

# T/HIS

## Version 12.0



*Oasys* Ltd  
The Software house of Arup

For help and support from OASYS Ltd please contact:

## **UK**

Arup Group Ltd  
The Arup Campus  
Blythe Gate  
Blythe Valley Park  
Solihull  
West Midlands  
B90 8AE  
United Kingdom  
Tel: +44 (0) 121 213 3399  
Fax: +44 (0) 121 213 3302  
Email: [dyna.support@arup.com](mailto:dyna.support@arup.com)  
Web: [www.oasys-software.com/dyna](http://www.oasys-software.com/dyna)

## **China**

Arup  
39/F-41/F  
Huai Hai Plaza  
Huai Hai Road (M)  
Shanghai  
China 200031  
Tel: +86 21 3118 8875  
Fax: +86 21 3118 8882  
Email: [china.support@arup.com](mailto:china.support@arup.com)  
Web: [www.oasys-software.com/dyna/cn](http://www.oasys-software.com/dyna/cn)

## **India**

Arup  
Plot 39, Ananth Info Park  
Opp. Oracle Campus  
HiTec City  
Madhapur Phase II  
Hyderabad 500081  
India  
Tel: +91 40 4436 9797/98  
Email: [india.support@arup.com](mailto:india.support@arup.com)  
Web: [www.oasys-software.com/dyna](http://www.oasys-software.com/dyna)

or contact your local Oasys Ltd distributor

---

<b>Development History</b>	<b>0.1</b>
New Features for version 12.0	0.1
New Features for version 11.0	0.1
New Features for version 10.2	0.1
New Features for version 10.0	0.1
New Features for version 9.4.1	0.2
New Features for version 9.4	0.2
New Features for version 9.3	0.2
New Features for version 9.2	0.2
New Features for version 9.0	0.3
New Features for version 8.2	0.3
New Features for version 8.1	0.3
New Features for version 8.0	0.4
Text conventions used in this manual	0.4
<b>1 Introduction</b>	<b>1.1</b>
1.1 Program Limits	1.1
1.2 Running T/HIS	1.2
1.3 Command Line Options	1.4
<b>2 Using Screen Menus</b>	<b>2.1</b>
2.1 Basic screen menu layout	2.1
2.2 Mouse and keyboard usage for screen-menu interface	2.2
2.3 Dialogue input in the screen menu interface	2.4
2.4 Window management in the screen interface	2.4
2.5 Dynamic Viewing (Using the mouse to change views).	2.5
2.6 "Tool Bar" Options	2.6
<b>3 Graphs and Pages</b>	<b>3.1</b>
3.1 Creating Graphs	3.1
3.2 Page Size	3.2
3.3 Page Layouts	3.2
3.3.1 Automatic Page Layout	3.2
3.4 Pages	3.6
3.5 Active Graphs	3.6
<b>4 Global Commands and Pages</b>	<b>4.1</b>
4.1 Page Number	4.1
4.2 PLOT (PL)	4.1
4.3 POINT (PT)	4.2
4.4 CLEAR (CL)	4.2
4.5 ZOOM (ZM)	4.2
4.6 AUTOSCALE (AU)	4.2
4.7 CENTRE (CE)	4.2
4.8 MANUAL	4.2
4.9 STOP	4.2
4.10 TIDY	4.2
4.11 Additional Commands	4.3
<b>5 Main Menu</b>	<b>5.1</b>
5.0 Selecting Curves	5.1
5.1 READ Options	5.6
5.2 WRITE Options	5.30
5.3 Curve Manager	5.32
5.4 Model Manager	5.42
5.5 EDIT Options	5.44
5.6 LINE STYLES	5.50
5.7 Command / Session Files	5.58
5.8 IMAGE Options	5.62
5.9 OPERATE Options	5.67
5.10 MATHS Options	5.72
5.11 AUTOMOTIVE Options	5.73
5.12 SEISMIC Options	5.80
5.13 MACRO Options	5.82
5.14 FAST-TCF Options	5.84
5.15 TITLE/AXES/LEGEND Options	5.87
5.16 DISPLAY Options	5.96
5.17 SETTINGS	5.100
5.18 MEASURE	5.105
5.19 Curve Groups	5.109
5.20 GRAPHS	5.112
5.21 PROPERTIES	5.113
5.22 UNITS	5.118
5.23 The Javascript Interface	5.123
5.24 Datum Lines	5.129
<b>6 Other Options</b>	<b>6.1</b>

---

---

6.1 Tool Bar	6.1
6.2 Graph Tool Bar	6.9
6.3 CURVE INFORMATION	6.10
6.4 Curve Histories ...	6.11
6.5 Keyboard Shortcuts	6.15
6.6 Preferences	6.19
<b>7 FAST-TCF</b>	<b>7.1</b>
7.0 FAST-TCF OVERVIEW	7.1
7.1 FAST-TCF INTRODUCTION	7.2
7.2 PAGE / GRAPH LAYOUT AND SELECTION	7.8
7.3 INPUT SYNTAX TO LOAD OTHER FILES	7.9
7.4 INPUT FOR DATA EXTRACTION REQUESTS	7.10
7.5 UNITS	7.25
7.6 CURVE TAGS	7.27
7.6.4 Using Curve Numbers	7.28
7.7 CURVE GROUPS	7.29
7.8 PERFORMING FAST-TCF CURVE OPERATIONS	7.30
7.9 APPLYING EXTRA OPTIONS TO DATA REQUESTS	7.34
7.10 Setting properties for curves	7.35
7.11 Defining Datums	7.37
7.12 FAST-TCF IMAGE OUTPUT OPTIONS	7.39
7.13 Outputting curve properties to text files, variables and REPORTER	7.47
7.14 FAST-TCF CURVE OUTPUT	7.51
<b>APPENDICES</b>	<b>A.1</b>
APPENDIX A - LS-DYNA Data Components	A.2
APPENDIX B - T/HIS CURVE FILE FORMAT	B.1
APPENDIX C - T/HIS BULK DATA FILE FORMAT	C.1
APPENDIX D - FILTERING	D.1
APPENDIX E - INJURY CRITERIA	E.1
APPENDIX F - Curve Correlation	F.1
APPENDIX G - The ERROR Calculation	G.1
APPENDIX H - The "oa_pref" preference file	H.1
APPENDIX I - Windows File Associations	I.1
APPENDIX J - T-HIS JavaScript API global class	J.1
APPENDIX K - Typed Commands	K.1
<b>Installation organisation</b>	<b>L.1</b>
Version 12.0 Installation structure	L.1
<b>JaDe: The JavaScript debugger</b>	<b>M.1</b>
Viewing the script files and functions	M.1
Adding/removing breakpoints	M.1
Running the script	M.2
Printing the value of a variable	M.3
The call stack	M.4
Exceptions	M.5

---

# Development History

## New Features for version 12.0

### Description

Selecting element integration points and element nodal data values  
 Support for reading CSV files with TAB or SPACE field separators  
 Modified Clip function  
 Modified Rolling Average Function  
 WIFAC Curve Correlation Function  
 Measure  
 JavaScript "Datum" Class

### Manual Section

[5.1.1.7](#) and [7.4.4](#)  
[5.1.7](#)  
[5.9.6](#)  
[5.9.27](#)  
[5.11.26](#)  
[5.18](#)  
[Datum](#)

## New Features for version 11.0

### Description

Read data from DIAdem format data files  
 Read data from NASTRAN Punch files  
 Enhancements to curve table  
 New options for the ZERO function  
 New dB curve function  
 New dBA curve function  
 New Octave curve function  
 New Automatic curve group options  
 DATUM Lines  
 Added new Auto\_Blank option  
 FAST-TCF "style\_m" command for specifying curve styles by model  
 Added LSDA (binout) support for ELOUT\_DET, ELOUT\_SSD and ELOUT\_PDS  
 Added support for DCFAIL file  
 Added support for DISBOUT file  
 Added support for PLLYOUT file

### Manual Section

[5.1.11](#)  
[5.1.12](#)  
[5.3.4](#)  
[5.9.38](#)  
[5.9.39](#)  
[5.9.40](#)  
[5.9.41](#)  
[5.19](#)  
[5.24](#)  
[6.2.4](#)  
[7.12.2](#)  
[Appendix A](#)  
[Appendix A.22](#)  
[Appendix A.6](#)  
[Appendix A.28](#)

## New Features for version 10.2

### Description

Added support for Local X,Y,Z forces from NODFOR file.

### Manual Section

[Appendix A.15](#)

## New Features for version 10.0

### Description

New Curve History menu for viewing and modifying curves  
 New JavaScript capability for creating and modifying curves  
 New curve palette options for default curve colours  
 New FAST-TCF wildcard options for specifying curve tags  
 FAST-TCF - Outputting a range of curves to a file.  
 Added support for CPM\_SENSOR ASCII file  
 Added support for TRHIST ASCII file and LSDA data

### Manual Section

[6.4](#)  
[5.23](#) & [Appendix J](#)  
[5.6.6](#) & [5.17.3](#)  
[7.6.3](#)  
[7.14.1](#)  
[Appendix A16](#)  
[Appendix A.27](#)

## New Features for version 9.4.1

### Description

Added support for  
\*CONSTRAINED\_JOINT\_STIFFNESS\_TRANSLATIONAL

### Manual Section

[7.4.5](#) & [Appendix A.17](#)

## New Features for version 9.4

### Description

Automatic results extraction  
New MONotonic curve function  
New Normalise X axis values function  
New Pure Butterworth filter function  
New curve correlation function COR3  
New Inverse FFT function  
Curve Properties  
Units  
User Defined Shortcuts  
New FAST-TCF commands for setting curve properties  
New FAST\_TCF plot setup commands  
Added support for new DBFSI data components and DBSENSOR ASCII file.  
Added support for RELAX ASCII file.  
Added support for TPRINT ASCII file.  
Updated T/HIS curve file format

### Manual Section

[5.1.1.2](#)  
[5.9.18](#)  
[5.9.22](#)  
[5.11.6](#)  
[5.11.25](#)  
[5.12.10](#)  
[5.21](#)  
[5.22](#)  
[6.4](#)  
[7.10](#)  
[7.12.3.1](#)  
[Appendix A.25](#)  
  
[Appendix A.1](#)  
[Appendix A.4](#)  
[Appendix B](#)

## New Features for version 9.3

### Description

Multiple Graphs and Pages  
"Quick Pick" Curve Options.  
New "Screen" option for creating curves interactively  
Floating legend  
User controlled formatting of Axis values.  
Interactive curve editing  
PNG and GIF image output formats  
New Postscript Driver  
PDF output option  
2 and 4 times screen resolution output options.  
Background images  
Extended colour palette plus user defined colours  
Enhanced FAST-TCF options  
Curve Groups  
New Curve Correlation Function  
Setting File  
Added support for shell and solid strain data components from LSDA file  
Added support for Airbag Part data components from LSDA file

### Manual Section

[3.1](#) & [5.20](#)  
[6.1.8](#)  
[5.1.8](#)  
[5.15.5](#)  
[5.15.2](#)  
[5.5](#)  
[5.8.1.1](#)  
[5.8.2](#)  
[5.8.2](#)  
[5.8.1.2](#)  
[5.8.3](#)  
[5.6.2](#)  
[7.0](#)  
[5.19](#)  
[5.11.23](#) & [Appendix F](#)  
[5.17.4](#)  
[Appendix A.5](#) & [Appendix A.7](#)  
[7.4.5](#) & [Appendix A.16](#)

## New Features for version 9.2

### Description

New menu interface

### Manual Section

[3.1](#)

Read MODEL data	<a href="#">5.1.1</a>
User defined preferences for LS-DYNA data sources	<a href="#">5.17</a>
Multiple Models	<a href="#">5.1.1.5</a> & <a href="#">5.4</a>
Enhanced FAST-TCF scripting (all data components and T/HIS curve functions)	<a href="#">7.0</a>
MACRO Functions	<a href="#">5.13</a>
Automatic FAST-TCF Script Generation	<a href="#">5.14.1</a>
Interactive FAST-TCF playback	<a href="#">5.14.2</a>
"Right-click" curve operations from within graphics window	<a href="#">6.3</a>
Enhanced interactive control of graph fonts, colours and text size	<a href="#">5.15</a>
Embedded preference editor	<a href="#">6.6</a>
Extended oa_pref options	<a href="#">Appendix H</a>

## New Features for version 9.0

Description	Manual Section
Graphics Box Options and Screen Layout	<a href="#">3.7</a>
Multiple Data Component Selection	<a href="#">5.1.0</a>
Improved "Curve Operations"	
New Maths Commands LOG(x), LOG10(x)	<a href="#">5.10.6</a> & <a href="#">5.10.7</a>
Show HIC / 3ms Clip	<a href="#">5.11.8</a> & <a href="#">5.11.10</a>
New Automotive Commands HIC(d), NIJ, TTI	<a href="#">5.11.9</a> , <a href="#">5.11.15</a> , <a href="#">5.11.16</a> , <a href="#">Appendix E.6</a> & <a href="#">Appendix E.7</a>
Curve Style manipulation	<a href="#">5.6</a> & <a href="#">Appendix B</a>
Curve Title and axis manipulation	<a href="#">5.12.8</a>
Curve tags	<a href="#">5.15</a> & <a href="#">Appendix B</a>
Working with more than 1000 curves	<a href="#">5.14</a>
LSDA file reading	<a href="#">5.1.1</a>
SETTINGS menu	<a href="#">5.17</a>
New Preference file options	<a href="#">Appendix H</a>
Fast TCF: Fast, automated LS-DYNA results extraction and plotting using T/HIS	<a href="#">7</a>

## New Features for version 8.2

Description	Manual Section
Curve reordering	<a href="#">5.0.1</a>
New Line Styles for colour plots	<a href="#">5.6.5</a>
Read KEYWORD option	<a href="#">5.1.5</a>
Read CSV option	<a href="#">5.1.8</a>
Write NASTRAN TABLE D1 option	<a href="#">5.2.1</a>
Write CSV option	<a href="#">5.2.1</a>
User defined Grid Intervals	<a href="#">5.15.2</a>
Multiple Y Axis	<a href="#">5.15.4</a>
New plot formats	<a href="#">5.15.5.4</a>
Improved COMbine function	<a href="#">5.9.7</a>
Improved Automotive filter functions	<a href="#">5.11</a>
New Preference file options	<a href="#">Appendix H</a>

## New Features for version 8.1

Description	Manual Section
Support for DEFORC ASCII file	<a href="#">5.1.1</a> & <a href="#">Appendix A</a>
Support for NODOUT ASCII file	<a href="#">5.1.1</a> & <a href="#">Appendix A</a>

Support for RCFORC ASCII file	<a href="#">5.1.1</a> & <a href="#">Appendix A</a>
Support for SBTOUT ASCII file	<a href="#">5.1.1</a> & <a href="#">Appendix A</a>
Automatic Labelling of Maxima & Minima	<a href="#">5.16.11</a>
New Operate Commands (ERR, WINDOW, MIN, MAX, AVE)	<a href="#">5.9</a> & <a href="#">Appendix G</a>
New Automotive Commands (Acceleration Severity Index, ASI)	<a href="#">5.11.13</a> & <a href="#">Appendix E</a>
New Automotive Commands (Theoretical Head Impact Velocity, THIV)	<a href="#">5.11.14</a> & <a href="#">Appendix E</a>
Command line options & Windows file associations	<a href="#">Appendix I</a>

## New Features for version 8.0

Description	Manual Section
Maximum number of points increased to 2,500,000	<a href="#">1.1</a>
BLANK VISIBLE option	<a href="#">5.2</a>
REVERSE ALL option	<a href="#">5.2</a>
Screen Picking Curves	<a href="#">5.0.1</a>
Support for RWFORC ASCII file	<a href="#">5.1.1</a> & <a href="#">Appendix A</a>
New Automotive Command (Exceedence plot, EXC)	<a href="#">5.11.11</a>
New Automotive Command (Viscous Criteria, VC)	<a href="#">5.11.12</a> & <a href="#">Appendix E</a>
New Seismic Commands (Baseline Correction, BLC)	<a href="#">5.12.12</a>

## Text conventions used in this manual

### Typefaces

Three different typefaces are used in this manual:

Manual text	This typeface is used for text in this manual.
<b>Computer type</b>	This one is used to show what the computer types. It is also used for equations, keywords (eg <b>*PART</b> ) etc.
Operator type	This one is used to show what you must type.
<b>Button text</b>	This one is used for screen menu buttons (eg <b>APPLY</b> )

### Notation

Triangular, round and square brackets have been used as follows:

- **Triangular**  
To show generic items, and special keys. For example: <list of integers> <filename> <data component><return> <control Z> <escape>
- **Round**  
To show optional items during input, for example: <command> (<optional command>) (<optional number>)

And also to show defaults when the computer prompts you, eg:

Give new value (10) :

Give model number (12) :

- **Square**  
To show advisory information at computer prompts, eg

Give filename: [.key] :

THIS >>> [H for Help] :



# 1 Introduction

T/HIS is an x/y plotting program, specifically written to perform two functions:

1. To produce time-history plots from transient analyses, such as those performed using LS-DYNA.
2. To plot any form of x/y data that is produced either by a program or by directly typing in values.

T/HIS is a graphically driven, interactive program. Input and manipulation of data is through a graphical user interface on systems capable of running X-Windows applications; selections are made through "pressing buttons" using a mouse. On machines not capable of running X-Windows it is also possible to use T/HIS in a "command line" mode of operation; instructions are entered through the keyboard to perform the required operations.

## 1.1 Program Limits

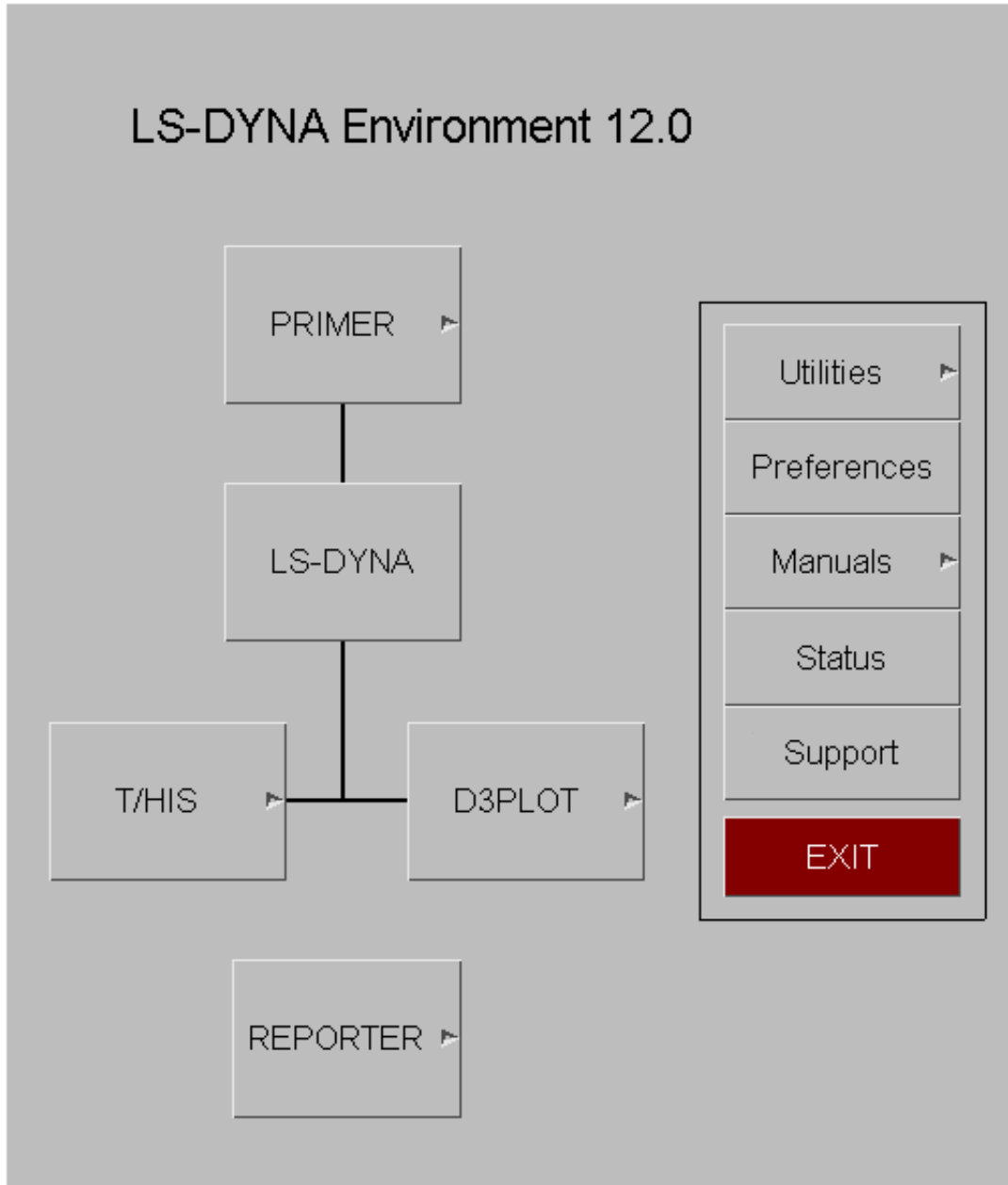
There are a number of limits in T/HIS of which the user should be aware. These are listed below:

<b>Number of graphs</b>	T/HIS can have a maximum of 32 graphs
<b>Number of curves</b>	The number of curves is unlimited
<b>Number of points</b>	The number of points that can be defined per curve is unlimited.
<b>Time-history blocks</b>	<p>In the interface to the LS-DYNA time-history (<b>.thf</b>) file there is a limit of 100,000 items in each of the node, solid, beam, shell and thick shell time-history blocks: thus 500,000 items overall.</p> <p>In the interface to the LS-DYNA extra time-history (<b>.xtf</b>) file up to 100,000 nodal reactions (or groups of reactions) may be processed.</p>
<b>Number of colours</b>	<p>By default, T/HIS curves wrap around the following six colours in order:</p> <p>WHITE RED GREEN BLUE CYAN MAGENTA</p> <p>However, a further 24 predefined colours are available if required and 6 user defined ones can be created.</p>
<b>Title</b>	The title can contain up to 80 characters.
<b>Labels</b>	Labels for axes and lines can contain up to 80 characters.

## 1.2 Running T/HIS

### 1.2.1 Starting the code

For users on a device with a window manager T/HIS is run from the **T/HIS** button in the SHELL:



If your system has been customised locally you may have to use some other command or icon: consult your system manager in this case.

### 1.2.2 Graphics Driver and Platforms

T/HIS 9.3 onwards use a OpenGL graphics driver.

Both the 32 and 64 bit versions of T/HIS use 32bit (single precision) numbers to store and plot data. The 32 bit version is limited to a maximum of 4GB of memory on all platform (3GB on windows).

### 1.2.2.1 "Batch" Mode

T/HIS 12.0 can run in "batch" mode where the main application window is not displayed on the screen. "Batch" mode is available on all platforms.

To start T/HIS in batch mode use the command line option "-batch".

**e.g. `this10.exe -tcf=script.inp -batch`**

When running in "batch" mode T/HIS will automatically exit at the end of the script regardless of whether or not "-exit" is specified.

**NOTE:** All image, postscript and PDF outputs require a DISPLAY on UNIX / LINUX systems. If you are running T/HIS in "batch" mode as part of a automatic post processing script then T/HIS must have a X Windows DISPLAY even though the main window is not displayed. If the machine you are using is a server or part of a cluster without an X-Server then T/HIS can be used with the Xvfb software.

### 1.2.3 Selecting a device when a window manager is not running

If you are running on a non-window device, for example a Tektronix display or emulator, you may not be able to use screen menus. Instead you will have to run in "command-line" mode.

It is very unlikely that a user on a modern workstation will see these options, since the machine will have a window manager and will be running in "screen menu" mode. If they do appear it suggests that the machine and/or software are wrongly set up: see 1.2.4 below for suggested remedies.

### 1.2.4 If T/HIS will not start in "screen-menu" mode

You may be running on a device with a window manager, but still only get the command-line prompt (and probably no menu driven \_93 shell either).

This is almost certainly because of one or both of the following setup errors:

- (1) The DISPLAY environment variable has not been set up, or has been set incorrectly. This tells the X11 window manager where to place windows, and it must be set to point to your screen. Its generic setup string is:

**setenv DISPLAY <hostname>:<display number>** (C shell syntax)

Where <hostname> is your machine's name or internet address, for example:

**setenv DISPLAY :0** (Default display :0 on this machine)

**setenv DISPLAY tigger:0** (Default display :0 on machine "tigger")

**setenv DISPLAY 69.177.15.2:0** (Default display :0, address 69.177.15.2)

You may have to use the raw network address if the machine name has not been added to your **/etc/hosts** file, or possibly the "yellow pages" server hosts file.

- (2) Your machine (strictly the X11 "server") has not been told to accept window manager requests from remote machines. This is usually the case when you are trying to display from a remote machine over a network, and you get the message similar to:

**Xlib: connection to "<hostname>" refused by server**

**Xlib: Client is not authorised to connect to server**

In this case go to a window with a Unix prompt on your machine, and type:

**xhost +**

Which tells your window manager to accept requests from any remote client. It will produce a confirmatory message, which will be something like:

**access control disabled, clients can connect from any host**

If T/HIS still fails to work then please contact your system manager, or contact Oasys Ltd for advice and help.

## 1.2.5 Command Line Mode

Command line mode is the main method of data input on non X-Windows devices. Command line mode is also available within the X-Windows screen interface and is accessed through the dialogue window. In command line mode the user will be presented with a prompt which also indicates which level of the menu structure the user is at. For example:

**Defaults >**

In response to the prompt a valid option must be given. These are usually a two or three letter abbreviation of a command; for example **PL** is the command to plot a graph. A list of the commands available is provided by typing **M** (for Menu). In addition to commands specific to one menu there are a number of commands which have the same effect throughout T/HIS.

**Q** - (Quit) Abort and return to current menu

**!** - Go up a level in the menu structure

**/** - Return to the top level menu

**;** - Equivalent to a **<carriage return>** in a string of commands

**M** - Lists menu.

Several commands can be strung together on one line, separated by spaces, for example:

**/DE GR ON**

Numeric data can also be included in the command line if required, for example:

**/OP ADX #1 7.2 #**

Commands can be in upper or lower case.

As well as menu level commands you will be asked questions such as:

**THF file to read (filename\_1)?**

The default response, if one exists, is given in parentheses.

## 1.3 Command Line Options

Instead of starting T/HIS using the Command shell it is also possible to start T/HIS from the command line with a number of optional input parameters. Starting T/HIS from the command line offers a number of advantages.

- Faster start-up is possible by pre-selecting the device type.
- The input filename can be specified and opened automatically.
- Faster start-up is possible by pre-selecting the device type

Argument format:

**<application name> (<arg 1>) ... (<arg n>) (<input filename>)**

T/HIS 12.0 can be started with a number of optional command line options

Graphics device type	<b>-d=&lt;device type&gt;</b>	Valid device types are:	
	eg <b>-d=default</b>	<b>opengl</b>	OpenGL
		<b>tty</b>	No windows
		<b>default</b>	OpenGL

Command file name	<b>-cf=&lt;filename&gt;</b> eg <b>-cf=run 1.tcf</b>	Any valid T/HIS command file filename
FAST-TCF input file	<b>-tcf=&lt;filename&gt;</b> eg <b>-tcf=run 1.inp</b>	Any valid T/HIS FAST-TCF command file filename
Settings file	<b>-set=&lt;filename&gt;</b> eg <b>-set=this001.set</b>	Any valid T/HIS settings file
Javascript	<b>-js=&lt;filename&gt;</b> eg <b>-js=sort curve.js</b>	Any valid T/HIS JavaScript file
LS-DYNA Model	<b>&lt;filename&gt;</b> eg <b>run_1.thf</b>	Any filename from the analysis  This should be the last argument on the command line.
LS-DYNA Model list  Specify a file containing a list of models for T/HIS to automatically open.	<b>-model_list=&lt;filename&gt;</b> eg <b>-model_List=job_list</b>	The model list file should contain the full pathname of one file from each model that T/HIS should open. Each file should be on a separate line and it should be the first item on each line (other items separated with commas can be specified on the same line for use with REPORTER).
Model Database file  Specify the name of the default model database file.	<b>-mdb=&lt;filename&gt;</b> eg <b>-mdb=database.xml</b>	The model database file is an XML format file that contains information on where models are located along with a brief description of each model. The model database can be used to easily select multiple models..
T/HIS curve file  Specify a T/HIS curve file containing one or more curves for T/HIS to automatically open.	<b>-cur=&lt;filename&gt;</b> or <b>-curve=&lt;filename&gt;</b> eg <b>-cur=test.cur</b>	
T/HIS curve file list  Specify a file containing a list of curve files for T/HIS to automatically open.	<b>-curve_list=&lt;filename&gt;</b> or <b>-curve=&lt;filename&gt;</b> eg <b>-cur=test.cur</b>	The curve list file should contain the full pathname of each curve file that you want T/HIS to open. Each file should be on a separate line.
T/HIS bulk data file  Specify a T/HIS BDF file containing one or more curves for T/HIS to automatically open.	<b>-bdf=&lt;filename&gt;</b> eg <b>-bdf=test.cur</b>	
Automatically maximises the T/HIS window so that it occupies the full screen.	<b>-maximise</b>	
Read THF file	<b>-thf=&lt;yes/no&gt;</b>	
Read XTF file	<b>-xtf=&lt;yes/no&gt;</b>	
Read LSDA (binout) file	<b>-lsda=&lt;yes/no&gt;</b>	
Read ASCII files	<b>-ascii=&lt;yes/no&gt;</b>	
Specifying a custom "oa_pref" file.  This causes an extra, optional "oa_pref" file to be read.	<b>-pref=&lt;filename&gt;</b>	<filename> must be a valid "oa_pref" file.  If it has no path prefixed, the file is assumed to be in the OA_INSTALL directory. Any legal filename may be used.
Use ELOUT instead of ELOUTDET	<b>-use_elout=&lt;yes/no&gt;</b>	By default T/HIS uses the ELOUTDET part of the LSDA file in preference to ELOUT if the LSDA file contains both. This option can be used to force T/HIS to use the ELOUT data when reading Shell and ThickShell data as the ELOUT data can be in the global coordinate system instead of the element local coordinate system.
Specify a directory for T/HIS to start in.	<b>-start_in=&lt;directory&gt;</b>	Any valid directory

Set the width of the T/HIS graph window (in pixels)	<b>-xres=&lt;size&gt;</b> eg <b>-xres=800</b>	
Set the height of the T/HIS graph window (in pixels)	<b>-yres=&lt;size&gt;</b> eg <b>-yres=600</b>	
Run T/HIS without the console window.	<b>-noconsole</b>	Windows only.
Run T/HIS in "batch" mode where the main application window is not displayed on the screen.	<b>-batch</b>	For this option to work you must also specify a command file " <b>-cf=filename</b> " and the name of the PTF file to open.  This option will automatically set " <b>-exit</b> " so that D3PLOT terminates after playing the command file.
Redirect output from the console window to a file on Windows.  To redirect output on Unix/Linux use the shell redirection options (typically > for <stdout>, & for <stderr>)	<b>-eo=&lt;filename&gt;</b> <b>-eo</b> <b>-eo=default</b>	<b>-eo=&lt;filename&gt;</b> is designed for the user to suppress the console and redirect logfile output to the specified filename. In order to permit multiple sessions to coexist on the same machine the process id will be appended to the <name> part of the filename to give <name>_pid.<ext>.  If plain "-eo" or "-eo=default" are found then filename generation is automatic, and the first valid of:  %TEMP%\this_log_<pid>.txt %TMP%\this_log_<pid>.txt %HOMESHARE%\this_log_<pid>.txt %USERPROFILE%\this_log_<pid>.txt will be used.
Stop and exit after command file	<b>-exit</b>	

Some examples for T/HIS might be:

**pathname/this12.exe -d=x run\_2.thf** (Use device X, open a .thf file)

**pathname/this12.exe -d=tty cf=batch.tcf -exit** (No graphics, run command file)

Note that no spaces should be left in the syntax <arg>=<value>.

For example: "-d = x" is illegal.

Correct syntax is: "-d=x"

## 2 Using Screen Menus

### [2.1 Basic screen menu layout](#)

### [2.2 Mouse and keyboard usage](#)

### [2.3 Dialogue input](#)

### [2.4 Window management](#)

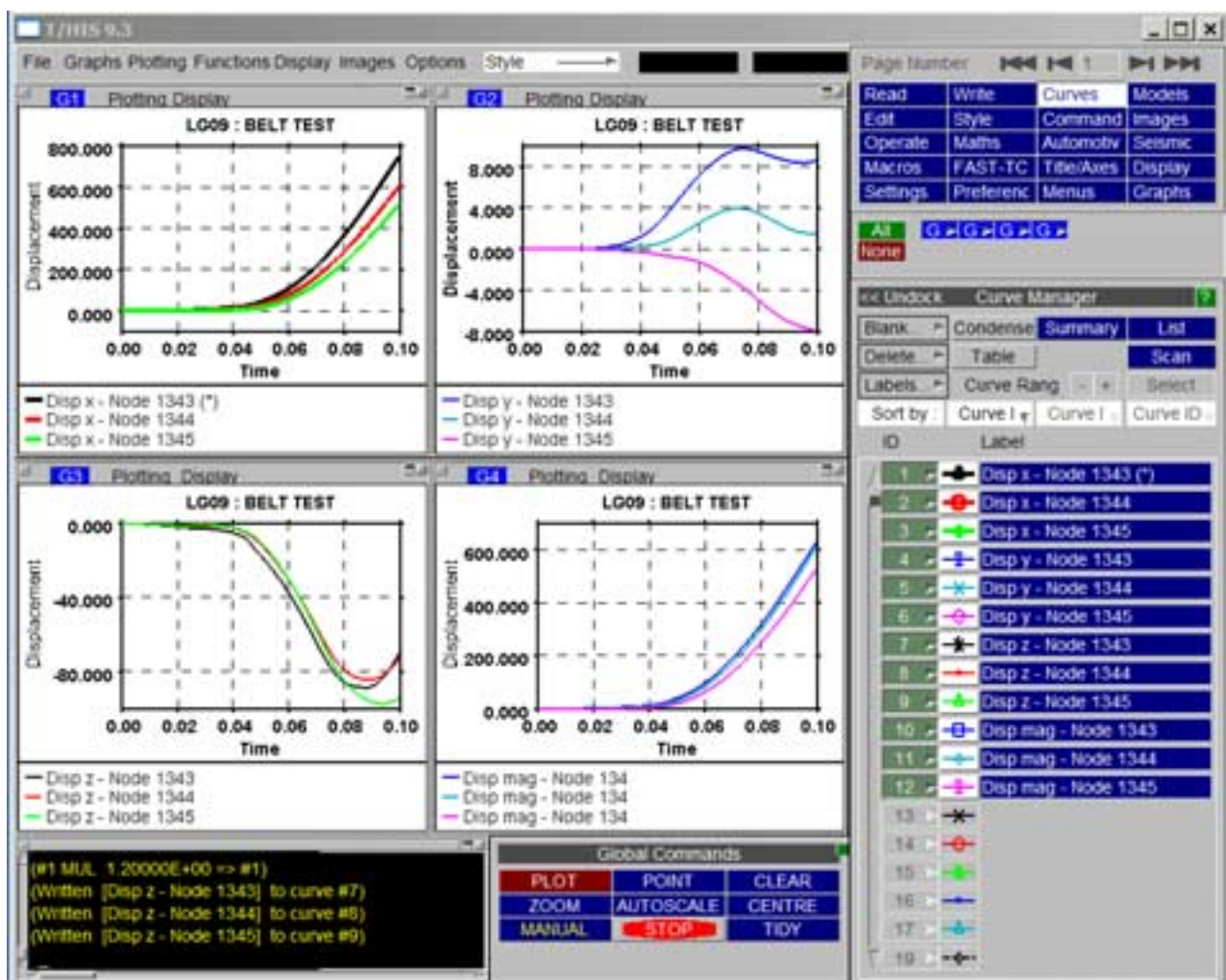
### [2.5 Dynamic Viewing \(Using the mouse to change views\)](#)

### [2.6 Graphics Box Options](#)

Versions of T/HIS prior to release 6.1 only had a "command-line" interface. This has been preserved for backwards compatibility, but a "screen-menu" interface has been added which allows you to drive the program almost entirely with the mouse.

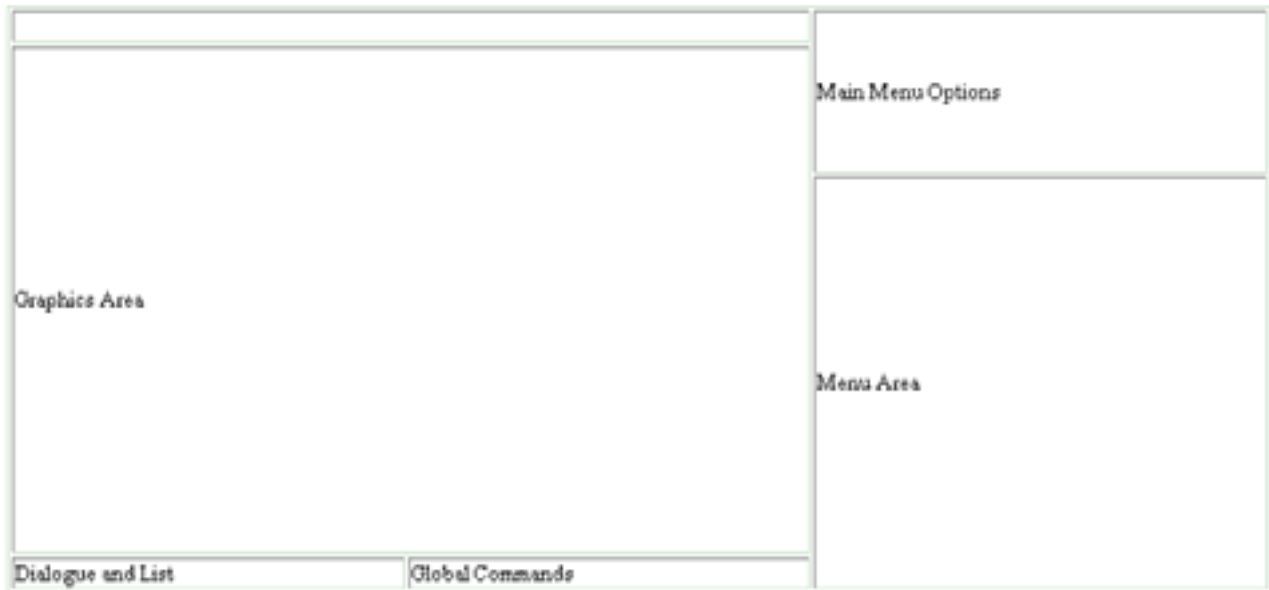
## 2.1 Basic screen menu layout

T/HIS runs within a single window, owned by the window manager, which has several sub-windows inside it. A typical T/HIS session will look like this:



The various sub-windows always exist within the master window, and may be moved and resized at will inside it. They will keep their relative size and position as the master window is changed in size and/or shape, and will reappear after the main window is de-iconised.

The default layout of the main sub-windows is as follows:



These windows cannot be dismissed. A brief description of their functions is:

- Main Menu Options** Provides access to the majority of the commands and options available in T/HIS through a series of sub menus (see [Section 6](#)).
- Graphics area** Is where graphs are drawn. In T/HIS 9.4 this area can contain a maximum of 32 graphs (see [Section 3](#)). Alternativley if graphs have been organised into pages (see [Section 3.3](#)) then this area will display a single page of graphs.
- Dialogue & list** Allows "command-line" input and output, also provides a listing area for messages.
- Menu Area** Displays the commands and options associated the current selection fromthe main menu options.
- Global Commands** Gives access to commonly used commands (see [Section 4](#)).

While you are free to reposition these master windows it is recommended that you keep to this default layout. This is because when further sub-windows appear their position and size is designed assuming this layout, and aims to obscure as little useful information as possible.

## 2.2 Mouse and keyboard usage for screen-menu interface

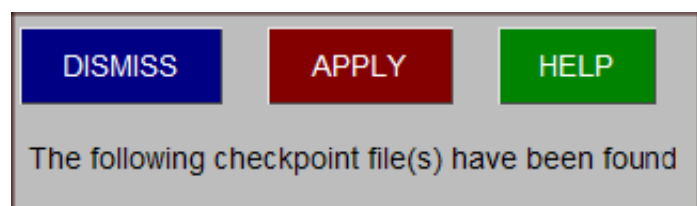
All screen-menu operations are driven with the left mouse button, with the following exceptions:

- (a) Text in the dialogue area and text boxes requires keyboard entry.
- (b) Text strings saved in the cursor "cut" buffer may be "pasted" into dialogue areas and text boxes using the middle mouse button.

The primitive "widgets" in the menu interface are used as follows:

### BUTTONS:

Screen buttons are depressed by clicking on them, but action only takes place when the mouse button is released, so it is safe to drag the (depressed) mouse around the screen.





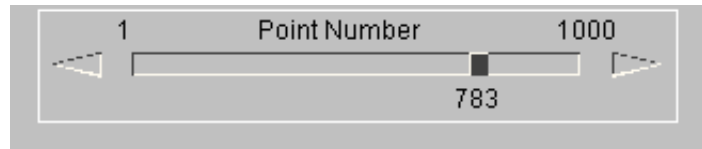
Buttons may also be greyed out to indicate that the option is not currently available. Buttons with "..." after them will usually invoke sub-menus.

**"Popup" window invocation:** Buttons with an ">" symbol may be selected normally with the left mouse button, but if the *right* mouse button is depressed over them it will invoke a "popup" window. Holding the right mouse button down move the cursor into this window to make a selection, or move elsewhere and release the button to deactivate the popup.



### SLIDERS:

Sliders are moved by clicking on the slider button itself, and then dragging it to a new position. They may also be moved automatically by clicking on, and holding down, one of the arrows at either end.



### TEXT BOXES:

Contact Tes

To enter text in a text box: first make it "live" by clicking on it, then type in text, then type **<return>** to enter the string. Clicking on a "live" box for a second time is exactly the same as typing **<return>**, so clicking twice on a box effectively enters its current contents. You can use the left and right arrow keys for line editing within a box: text entry takes place after the current cursor position.

### RADIO BOXES

A "radio" set is provided where only one selection is possible from a range of options. In this example the postscript laser output has been set to a single image per page.

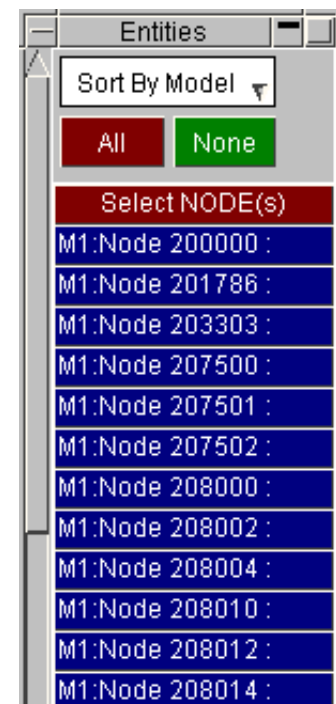


### MENU SELECTIONS:

Menus of items are used when you need to make one or more selections from a (potentially) long list. Click on the row you want to select: clicking on a row that is already selected will have the effect of unselecting it. When the list is too long to display in the window you can use the vertical scroll-bars to move up and down it.

A range of items may be selected by either

- 1) Click on the first item and hold down the mouse key, drag the mouse to the last item in the list. All items between the first and last including the first and last are selected.
- or
- 2) Click on the first item, hold down the SHIFT key and click on the last item in the list. All items between the first and last including the first and last are selected.



## 2.3 Dialogue input in the screen menu interface

The full command-line capability is preserved when T/HIS is running in screen-menu mode, and you are free to mix command-line and mouse-driven input at will. There are some situations in which command-line input is more efficient: for example when entering lists of explicit entities.

Commands are entered in the dialogue box:



As this example shows the dialogue box is also used for listing messages, warnings and errors to the screen. It can be scrolled back and forth (its buffer is 200 lines long) to review earlier messages. The following colours are used:

Normal messages and prompts	Yellow
Text typed in by you	White
Warning messages	Magenta
Error messages	Red


There is a minor limitation when mixing command-line and screen-menu mode: you cannot perform the same function simultaneously in both modes. If you attempt to do so you will get the message:


**WARNING: recursive access attempted**

And you will not be permitted to continue.

## 2.4 Window management in the screen interface

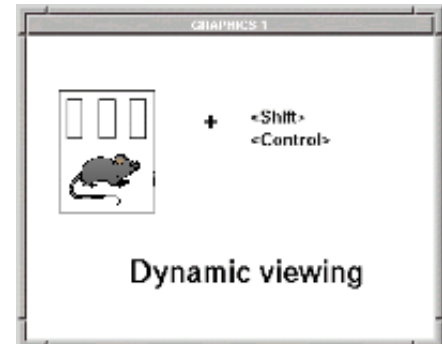
Moving, resizing and scrolling of windows is based on the conventions used in the Motif Window Manager.

- To move a window:** Click down on its title bar, then drag the window to where you want it to be. A "rubber-band" outline moves to show the window's current position.
- To resize a window:** Either
- Click on a border bar to move just that side, or on a corner bar to move both sides attached to that corner. Again, a rubber-band outline shows you the new shape.
- or
- Use the **MAXIMISE** button  in the top right hand corner of the window to increase the size of the window to the largest possible size.

- To scroll a window:** If a window has got too small for its contents then horizontal and/or vertical scrollbars will appear. Click on a scrollbar slider and move it to the desired position, the window contents will scroll as you do so. Alternatively click on the arrows at either end of the scrollbar for timed motion in that direction.
- To minimise a window:** Click on the button  in the top right hand corner of the window. When a window has been iconised it will appear in the **ICON** area at the bottom of the screen.
- To restore a window:** Iconised windows may be restored by clicking on the icon in the ICON area.

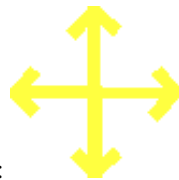
## 2.5 Dynamic Viewing (Using the mouse to change views).

"Dynamic" viewing is the name given to the process in which you perform viewing transformations by moving the mouse around the screen.



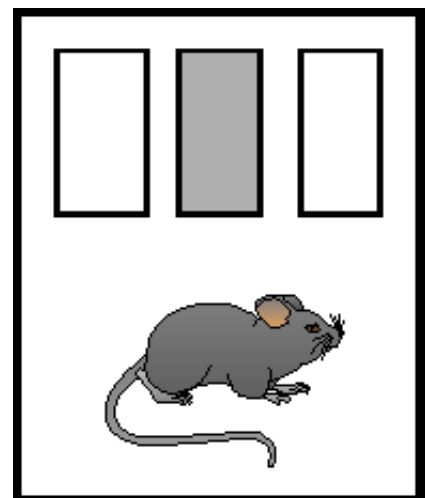
### 2.5.1 Dynamic Translation.

Dynamic translation uses **<mid mouse> + <left shift>**



The cursor symbol is yellow, and looks like:

The relationship between mouse and image motion is intuitive: the object tracks the mouse motion in the screen XY plane. The initial position of the mouse is irrelevant.



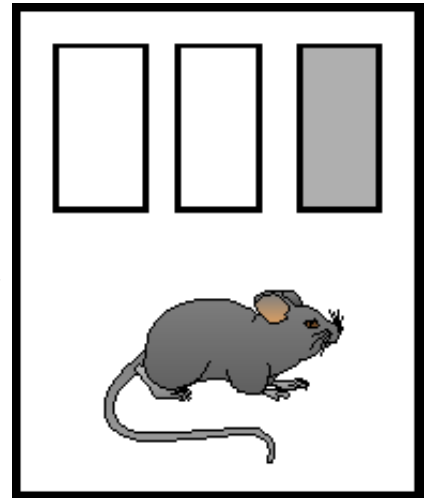
## 2.5.2 Dynamic Magnification (Scaling).

Dynamic scaling uses `<right mouse> + <left shift>`



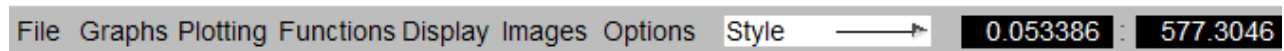
The cursor symbol is green, and looks like:

Mouse motion to the right and up makes the image larger, left and down smaller. The initial position of the mouse is irrelevant. A horizontal movement will scale just the x-axis while a vertical movement will scale just the y-axis.



## 2.6 "Tool Bar" Options

Across the top of the main graphics window are a number of buttons that can be used to access other T/HIS menus (see [Section 6.1](#)) for more details..



If the graphics box is [maximised](#) to take up the whole of the main window these buttons can be used to access the rest of the T/HIS menus without having to resize the graphics box between commands. Almost all of the options and functions in these menus may also be accessed from other menu locations, e.g. the Main Menu area.

## 3 Graphs and Pages

### [3.1 Creating Graphs](#)

#### [3.2 Page Size](#)

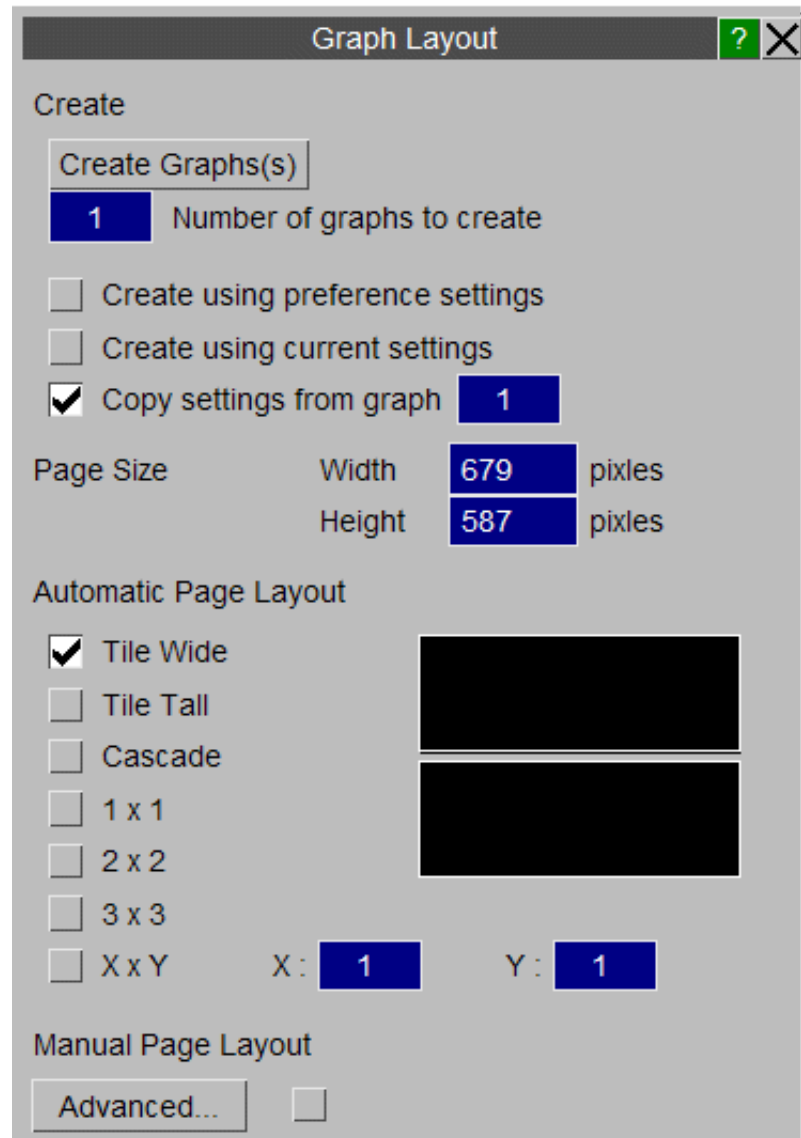
#### [3.3 Page Layouts](#)

#### [3.4 Pages](#)

#### [3.5 Active / Inactive Graphs](#)

T/HIS 9.4 can display a maximum of 32 graphs. Each graph can have a different appearance and they can display different curves.

Graphs can be laid out using a number of different formats and they can be organised into 'Pages'.



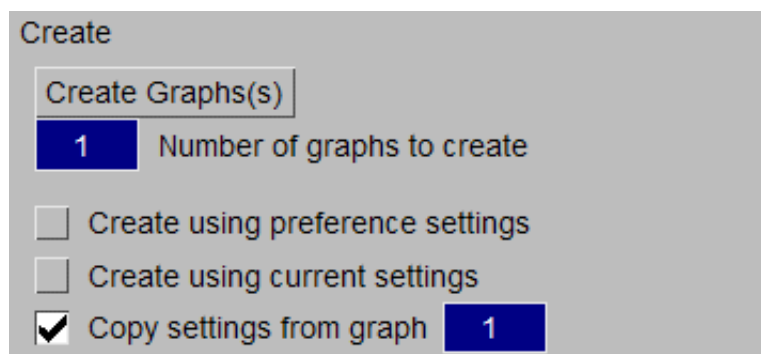
### 3.1 Creating Graphs

**Create Graphs** Create a new graph.

The [shortcut key](#) 'G' can also be used to create new graphs.

**Number of graphs to create**

This option can be used to create multiple graphs.



When new graphs are created the initial settings for each graph can be copied from 3 different sources.

- Create using preference settings**

The [Display](#) and [Axis Settings](#) are copied from the preference file.
- Create using current settings**

The [Display](#) and [Axis Settings](#) are copied from the current settings in the Display and Axis menus.
- Copy settings from graph n**

The [Display](#) and [Axis Settings](#) are copied from the specified graph.

### 3.2 Page Size

These options can be used to specify the total size of the area (in pixels) used by the graph windows.

Page Size	Width	679	pixels
	Height	587	pixels

### 3.3 Page Layouts

#### 3.3.1 Automatic Page Layout

If an Automatic page layout is used and the layout is set to [1 x 1](#), [2 x 2](#), [3 x 3](#) or [X x Y](#) T/HIS will automatically create multiple pages and position the graphs on each page if required.

Automatic Page Layout

☒ Tile Wide

☐ Tile Tall

☐ Cascade

☐ 1 x 1

☐ 2 x 2

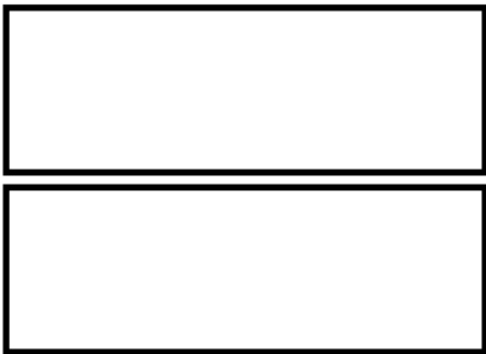
☐ 3 x 3

☐ X x Y

X : 1

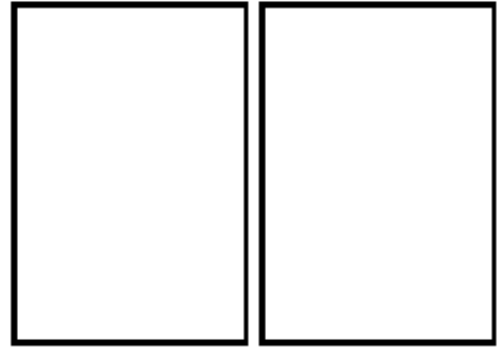
Y : 1

**Tile Wide**  
All of the graphs are positioned on a single page.

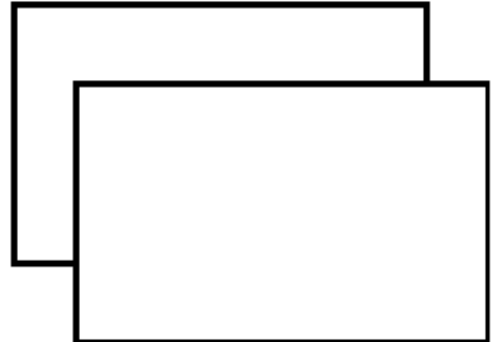


**Tile Tall**

All of the graphs are positioned on a single page.

**Cascade**

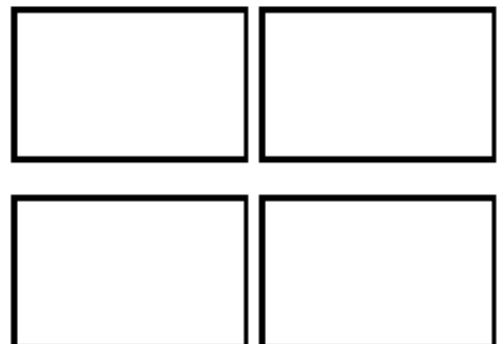
All of the graphs are positioned on a single page

**1 x 1**

Each graph is positioned on it's own page.

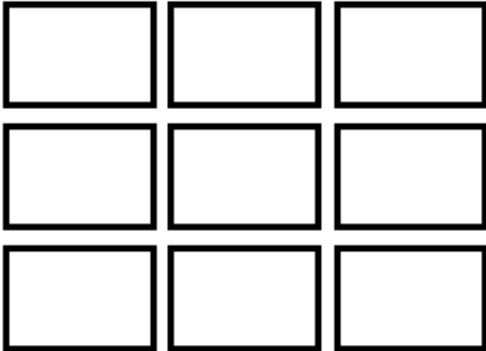
**2 x 2**

Graphs are arranged in a 2 by 2 grid. If there are more than 4 graphs then graphs 1 to 4 are positioned on page 1, 5 to 8 on page 2 ...



3 x 3

Graphs are arranged in a 3 by 3 grid. If there are more than 9 graphs then graphs 1 to 9 are positioned on page 1, 10 to 18 on page 2 ...



X x Y

Graphs are arranged in a X by Y grid.

3.3.2 Manual Page Layout

Manual page layout can be used to give more control over which graphs appear on which page. Unlike the Automatic page layouts a graph can appear on more than one page.

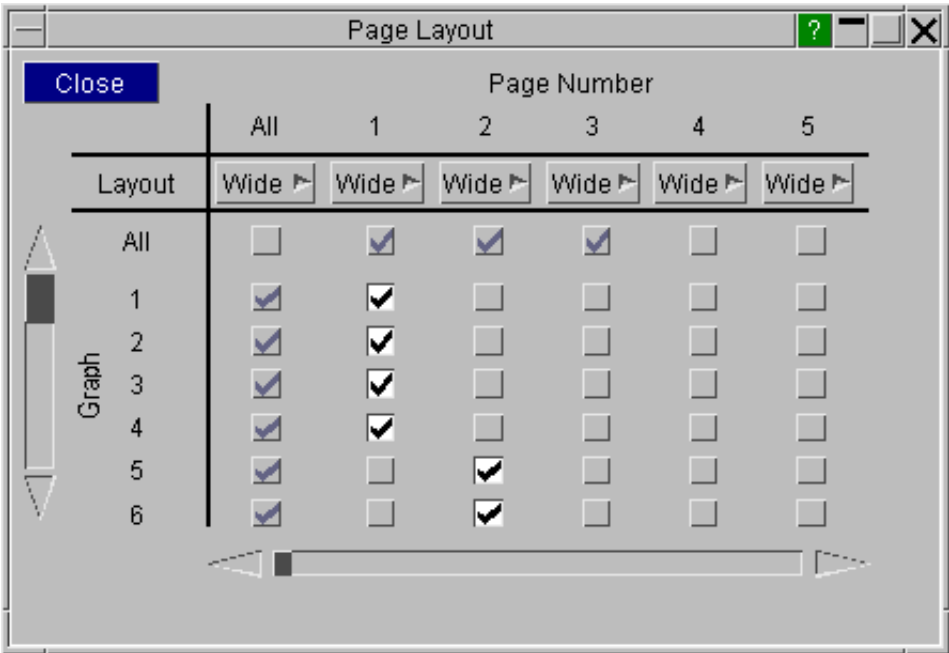


Advanced

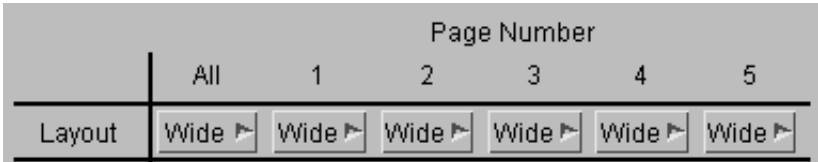
The Advanced option displays the Page Layout menu.

This menu can be used to select which graphs appear on each page. Each graph can appear on more than one page.

A range of graphs can be added/removed from pages by selecting the first graph/page combination and then holding down **SHIFT** while selecting the second graph/page.



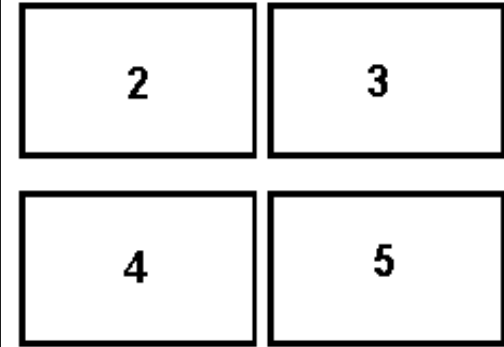
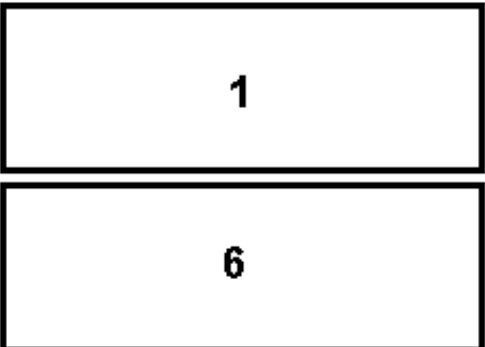
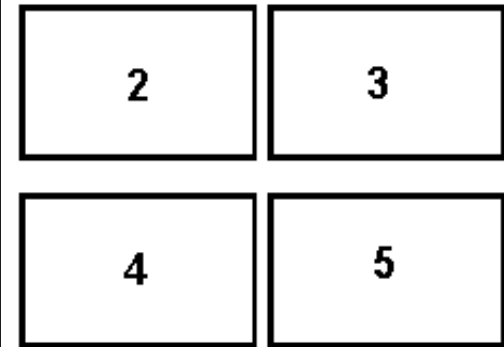
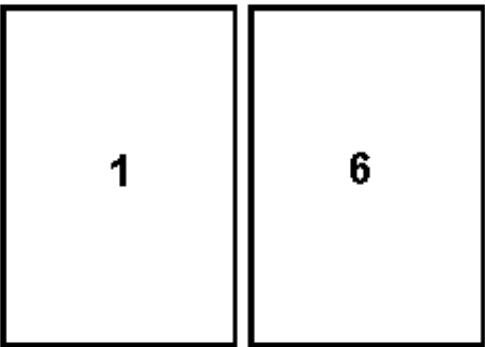
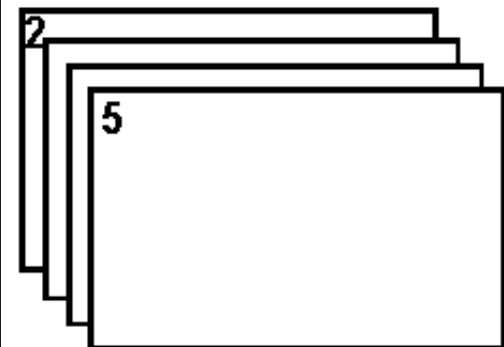
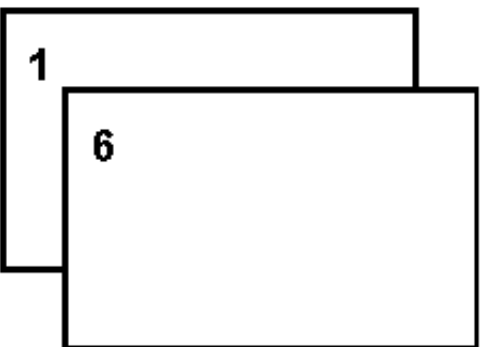
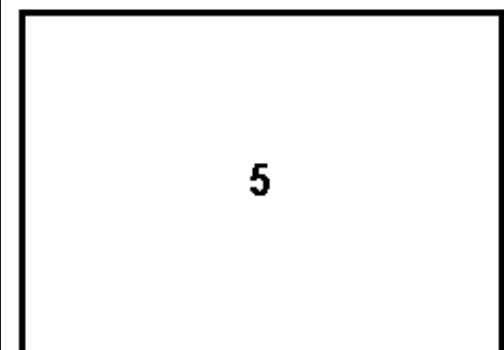
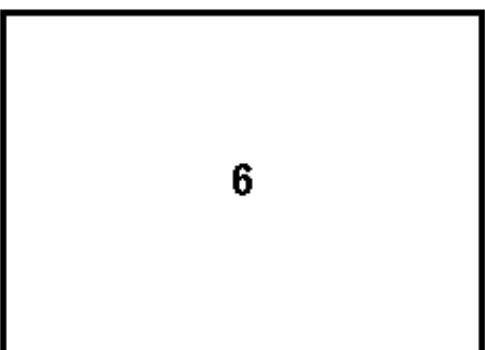
Each page can have a different layout or they can all be the same.





With the Advanced option the **Graph Layout** options work in exactly the same way as the [Automatic Page Layout](#) options, except they only position the graphs defined on each page.

If for example T/HIS has 6 graphs defined and graphs 2,3,4,5 are defined on page 1 and graphs 1 and 6 are on page 2 then the different graph layout options would produce the following.

	Page 1	Page 2
<b>Tile Wide</b>		
<b>Tile Tall</b>		
<b>Cascade</b>		
<b>1 x 1</b> (stacked on top of each other)		

<b>2 x 2</b>	<div> <div>2</div> <div>3</div> <div>4</div> <div>5</div> </div>	<div> <div>1</div> <div>6</div> </div>
<b>3 x 3</b>	<div> <div>2</div> <div>3</div> <div>4</div> <div>5</div> </div>	<div> <div>1</div> <div>6</div> </div>
<b>X x Y</b>	Layout depends on X and Y	Layout depends on X and Y

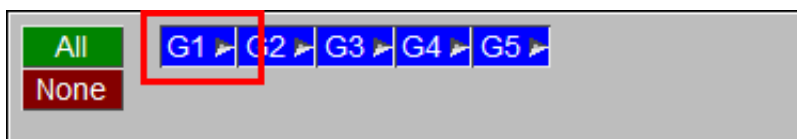
## 3.4 Pages

T/HIS can have a maximum of 32 pages, each page can contain multiple graphs. For more information on selecting the currently displayed page [see Section 4.1](#). The [Image Output](#) options and the [FAST-TCF Create](#) option can produce output for either a single page or multiple pages if graphs are located on more than one page.

## 3.5 Active Graphs

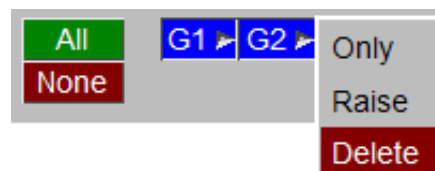
If T/HIS contains more than one graph then each graph can be toggled between being active or inactive.

All the graphs can be activated using the **All** button or deactivated using the **None** button.

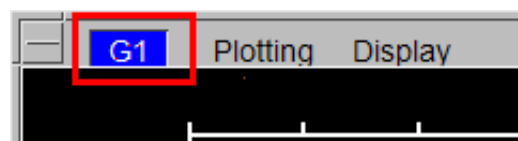


There is a popup menu attached to each button that can be used to select that graph **Only**, **Raise** the graph so that it is in front of any others or to **Delete** the graph.

When a graph is deleted any graphs with higher numbers are renumbered downwards to remove any gaps in the graph numbering.



Graphs can also be activated / deactivated using the button located in the top left hand corner of each graph.



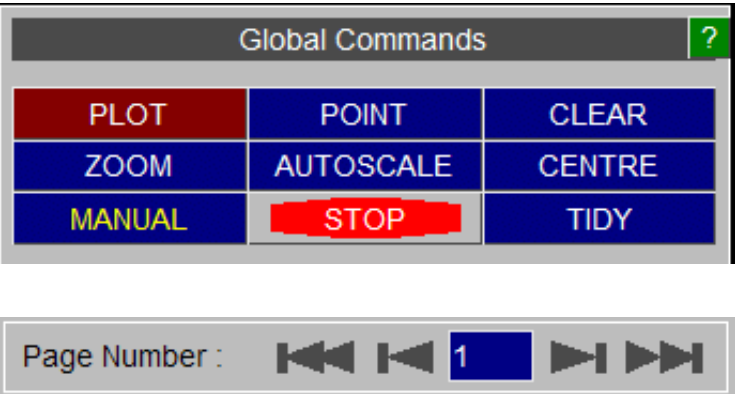
The options in the [Display](#) and [Title/Axes](#) menus that control the appearance of graphs are only applied to active graphs.

When new curves are created by reading in data from files the new curves are automatically unblanked in all of the currently active graphs and blanked in any inactive graphs.



# 4 Global Commands and Pages

- [4.1 Page Number](#)
- [4.2 PLOT](#)
- [4.3 POINT](#)
- [4.4 CLEAR](#)
- [4.5 ZOOM](#)
- [4.6 AUTOSCALE](#)
- [4.7 CENTRE](#)
- [4.8 MANUAL](#)
- [4.9 STOP](#)
- [4.10 TIDY](#)
- [4.11 Additional Commands](#)



The following commands are to be found as buttons on the **GLOBAL MENU** panel. (The command line codes are given in parentheses.)

All of the commands in the GLOBAL MENU can also be accessed via the **PLOTTING** button at the top of the graphics window.

## 4.1 Page Number

If T/HIS contains more than one graph ([see section 3.1](#)) then the graphs can be positioned on seperate Pages within T/HIS. This menu can be used to select a specific page or it can be used to step through the pages one by one.

	Shortcut Key	
	Goto Page 1	Home
	Go back 1 Page	Page Up
	Goto Page (n)	N/A
	Go back 1 Page	Page Down
	Goto Page 32	End

## 4.2 PLOT (PL)

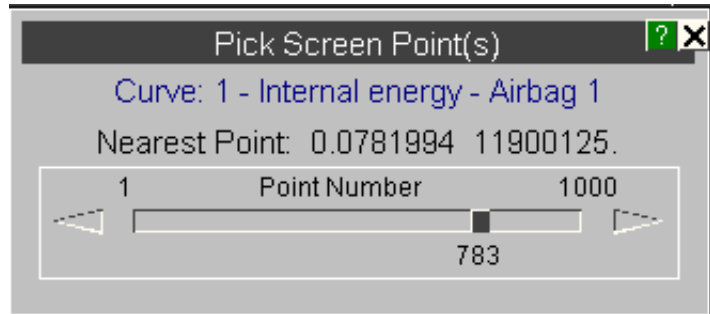
This option will plot all the curves that are currently UNBLANKED (see [Section 5](#)).

## 4.3 POINT (PT)

When selected this option waits for the user to pick a point in the main graphics screen.

Once a point has been picked the <x> and <y> values of the point picked are reported along with the ID of the nearest curve and the nearest point on that curve.

After a point has been selected on the screen the slider may be used to move to other points along the same curve.



## 4.4 CLEAR (CL)

Clears the graphics screen.

## 4.5 ZOOM (ZM)

The cursor appears on the screen and may be used to select the required plot area by choosing opposite corners of a box. The graphs are then replotted. Using **ZOOM** implicitly turns autoscaling off.

## 4.6 AUTOSCALE (AU)

Autoscales the plot size for all current unblanked curves in the graphics window and re-displays the plot.

## 4.7 CENTRE (CE)

Pick a point on the screen using the cursor to be the new plot centre. It affects the x/y offsets but not the scales.

## 4.8 MANUAL

Displays the online (HTML) version of the manual

## 4.9 STOP

Some operations, like reading a file containing many curves in to T/HIS, can take a long time. This button can be used to stop some long operations without having to exit from T/HIS.

## 4.10 TIDY

This option can be used to reset the menu layout to the default settings.

## 4.11 Additional Commands

A number of additional global commands exist in command line mode. These functions exist in screen menu mode within other menu levels.

- (**PF**) Creates a postscript plot file. Either A4 landscape or A4 portrait formats may be chosen. A title and figure number are also requested. Other plot setting may be made in the command line mode **UTILITIES** menu.
- (**BL**) Blank a currently displayed curve.
- (**UB**) Unblank a curve that has been blanked.
- (**RM**) Remove (delete) a curve. Once a curve has been removed it is lost from the system.
- (**ER**) Erase (delete) all existing curves from T/HIS. (Equivalent to the command **RM \* .**)
- (**GS**) Global status: displays the current number of curves, their labels and whether they are blanked.
- (**CO**) Condense: renumbers all curves to fill any gaps in curve numbers.
- (**LM**) Gives the current program limits.
- (**FT**) File tracking: lists the 20 files which have been accessed most recently by T/HIS, giving details of the type of file and whether it was read from or written to.
- (**EX**) Exits (leaves) the program.





## 5 Main Menu

### [5.1 READ Options](#)

### [5.2 WRITE Options](#)

### [5.3 CURVE Manager](#)

### [5.4 MODEL Manager](#)

### [5.5 EDIT Options](#)

### [5.6 STYLE Menu](#)

### [5.7 Command File](#)

### [5.8 IMAGE Options](#)

### [5.9 OPERATE Options](#)

### [5.10 MATHS Options](#)

### [5.11 AUTOMOTIVE Options](#)

### [5.12 SEISMIC Options](#)

### [5.13 MACRO Options](#)

### [5.14 FAST-TCF Options](#)

### [5.15 TITLE/AXES Options](#)

### [5.16 DISPLAY Options](#)

### [5.17 SETTINGS Menu](#)

### [5.18 MEASURE Menu](#)

### [5.19 GROUPS Menu](#)

### [5.20 GRAPHS Menu](#)

### [5.21 PROPERTIES Menu](#)

### [5.22 UNITS Menu](#)

### [5.23 JavaScript Menu](#)

### [5.24 Datum Menu](#)

 Read	Write	Curves	Models
Edit	Style	Properties	Images
Operate	Maths	Automotive	Seismic
Macros	FAST-TCF	Title/Axes	Display
Settings	Measure	Groups	Graphs
Command File	Units	JavaScript	Datum

The **MAIN MENU** provides access to a number of separate menus that perform most of the operations available within T/HIS from reading in data to producing postscript laser files.

## 5.0 Selecting Curves

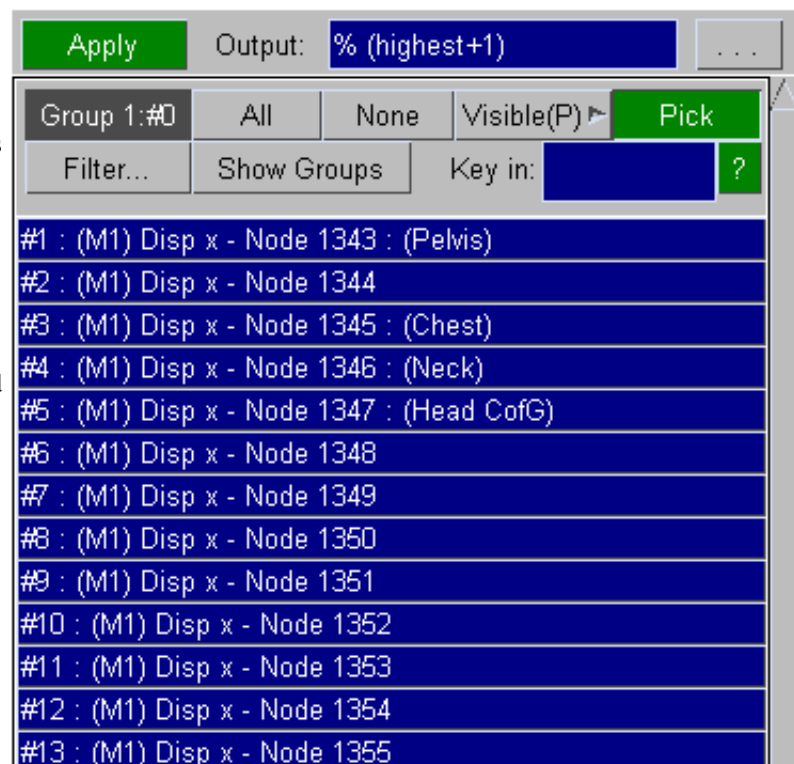
### 5.0.1 Input Curves

#### By Curve ID

A number of the menus require a range of curves to be selected. When a range of curves has to be selected a menu containing a list of the available curves will be displayed (see figure, right).

A range or curves may be selected by either

1. Click on the first item and hold down the mouse key, drag the mouse to the last item in the list. All items between the first and last including the first and last are selected.
2. Click on the first item, hold down the SHIFT key and click on the last item in the list. All items between the first and last including the first and last are selected.



**VISIBLE (P)age**

This option will select all of the curves that are unblanked in any graph on the current page.

Visible (Page) ▾

Visible on Page

Visible All Pages

Explain This

**VISIBLE (A)ll Pages**

This option will select all curves that are unblanked in at least one graph.

**PICK**

Alternatively curves may be picked from the screen. With this option the left mouse button is used to select curves while the right button deselects curves. As each curve is selected/deselected its name and number will be reported to the user and it will be highlighted on the screen.

A range of curves can be selected interactively by dragging out an area on the screen while holding down the left mouse button.

**FILTER...**

This option can be used to filter the list of curves displayed by model. When this option is selected a list containing all of the current models in T/HIS is displayed and the models can be selected or deselected. Any curves that belong to a deselected model will then be filtered out of the curve list.

**SHOW GROUPS**

This option will display a list of the currently defined curve groups instead of curves

## By Curve Group

In addition to selecting individual curves it is also possible to select curves by [Curve Group](#) if they have been defined.

- If a curve is defined in more than one group then it will be selected if at least one of the groups is selected.
- If more than one group containing the same curve is selected then the curve will only be counted once as an input curve.

Apply		Output: % (highest+1)		...
Group 1: #0	All	None	Visible(P) ▾	Pick
Filter...	Show Curves	Key in:		?
<b>CURVE GROUP LIST</b>				
#1 : Model_1				
#2 : Displacements				
#3 : Velocities				
#4 : Accelerations				

## By Command Line

In command line mode a single curve may be selected by typing in a range. A valid syntax is:

A single curve number	e.g. #27
A "from":"to" range	e.g. #10:#30 (no gaps, ":" mandatory)
A compound list in "(..)"	e.g. (#1 #2 #10:#30 #3 #97)

**In all contexts the order in which a group is defined does NOT influence the order in which it is processed. It is ALWAYS processed in ascending sequential order.**

Thus the addition operation

```
/OP ADD (#30 #20 #10) (#1 #2 #3) #40
```

will produce the results

```
#40 = #10 + #1
```

```
#41 = #20 + #2
```

```
#42 = #30 + #3
```

## 5.0.2 Output Curves

All operations that generate new curves must have a target curve defined. This must be one of the following:

#nnn	a specific curve number nnn
#	meaning "the lowest free curve"
%	meaning "the highest free curve"

In all cases output will start at the relevant curve number, however defined, and will rise sequentially with no gaps. This can cause an existing curve to be overwritten, or the output curve number to exceed the limit of 999. Both conditions are checked for: a warning is given if either will occur should the operation go ahead, and an opportunity given to modify or abort the pending operation.

There is a further output option that is only valid for operations where the input is a curve group:

- . meaning "overwrite the input curve(s)"

In this case the input curves are overwritten without warning. For example, this option might be used to integrate a set of curves, overwriting the original results with the integrated values.

Any curve number between 1 and 999 may be used as an input or output curve. It is not necessary to use curves sequentially; gaps are permitted in curve number usage. Therefore curves #1 and #10 can be used, for example, without having to use the intervening curves #2 to #9. Likewise, deleting a curve will no longer cause those above it to be renumbered downwards to fill the gap.

## 5.0.3 Curve Operations

The functions available fall into four distinct groups,

- 1) Separate functions involving two groups of curves, where the result is of the form:  
 $\langle R_n \rangle = \langle G_{1n} \rangle [OP] \langle G_{2n} \rangle$
- 2) Separate functions involving only one group of curves, where the result is of the form:  
 $\langle R_n \rangle = [OP] \langle G_{1n} \rangle$
- 3) Single output from only one group of curves, where the result is of the form:  
 $\langle R \rangle = [OP] \langle G_{1(1...n)} \rangle$
- 4) Separate functions involving three groups of curves, where the result is of the form:  
 $\langle R_n \rangle = \langle G_{1n} \rangle [OP] \langle G_{2n} \rangle [OP] \langle G_{3n} \rangle$

Currently the only function that has 3 curves groups as input is the VEC operation

### 1) Separate Functions On Two Groups

These functions display a menu in which **two** groups of curves may be selected, (see right).

You must define one or more curves in group #1, and group #2 must be:

either A group of as many curves as there are in group #1.

or A single curve. Every curve in group #1 is applied to this curve.

or A constant value, entered in the **Key in:** text box.

You can pick curves in either group from their menus, or type a range into the **Key in:** box.

**NOTE :** the order in which they are processed is ascending sequential, **not the order in which you define them.**

Apply		Output: % (highest+1)		...
Group 1: #0	All	None	Visible(P) ▾	Pick
Filter...	Show Groups		Key in:	?
#1 : (M1) Disp x - Node 1343 : (Pelvis)				
#2 : (M1) Disp x - Node 1344				
#3 : (M1) Disp x - Node 1345 : (Chest)				
#4 : (M1) Disp x - Node 1346 : (Neck)				
#5 : (M1) Disp x - Node 1347 : (Head CofG)				
#6 : (M1) Disp x - Node 1348				
#7 : (M1) Disp x - Node 1349				
#8 : (M1) Disp x - Node 1350				
#9 : (M1) Disp x - Node 1351				
Group 2: #0	All	None	Visible(P) ▾	Pick
Filter...	Show Groups		Key in:	?
#1 : (M1) Disp x - Node 1343 : (Pelvis)				
#2 : (M1) Disp x - Node 1344				
#3 : (M1) Disp x - Node 1345 : (Chest)				
#4 : (M1) Disp x - Node 1346 : (Neck)				
#5 : (M1) Disp x - Node 1347 : (Head CofG)				
#6 : (M1) Disp x - Node 1348				
#7 : (M1) Disp x - Node 1349				
#8 : (M1) Disp x - Node 1350				
#9 : (M1) Disp x - Node 1351				

## 2) Separate Functions On A Single Group

These functions display a menu in which one group of curves may be selected, (see right).

Operations apply separately and uniquely to each selected curve.

As before, the order of processing is ascending sequential, not the order in which you define them.

Apply		Output: % (highest+1)		...
Group 1: #0	All	None	Visible(P)	Pick
Filter...	Show Groups	Key in:		?
#1 : (M1) Disp x - Node 1343 : (Pelvis)				
#2 : (M1) Disp x - Node 1344				
#3 : (M1) Disp x - Node 1345 : (Chest)				
#4 : (M1) Disp x - Node 1346 : (Neck)				
#5 : (M1) Disp x - Node 1347 : (Head CofG)				
#6 : (M1) Disp x - Node 1348				
#7 : (M1) Disp x - Node 1349				
#8 : (M1) Disp x - Node 1350				
#9 : (M1) Disp x - Node 1351				
#10 : (M1) Disp x - Node 1352				
#11 : (M1) Disp x - Node 1353				
#12 : (M1) Disp x - Node 1354				
#13 : (M1) Disp x - Node 1355				

## 3) Single Output From A Single Group

These functions require a single group of curves as input like the functions above. The output is a single curve.

## 5.1 READ Options

T/HIS can **READ** data from a number of sources including LS-DYNA binary output files, LS-DYNA ASCII files and tabulated x/y data files. In addition this menu allows data for new curves to be entered directly using the keyboard.

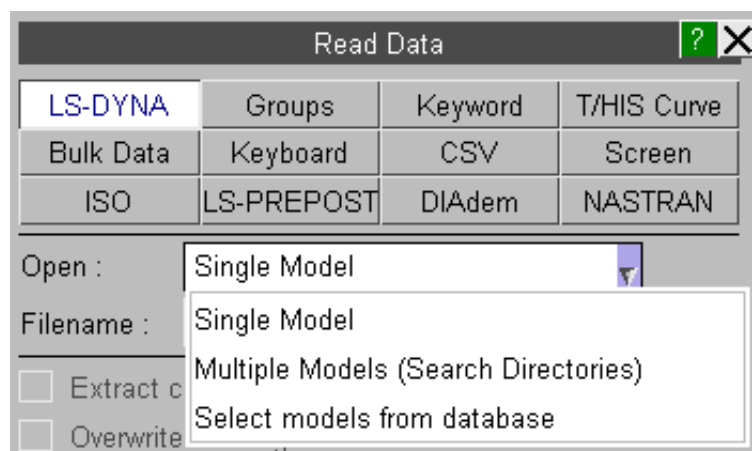
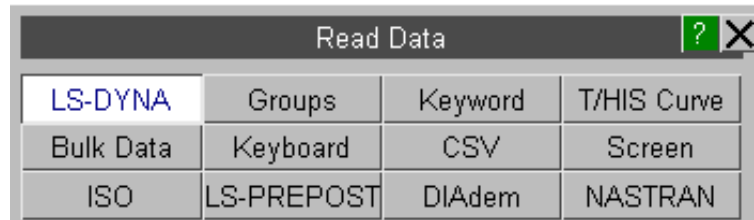
### 5.1.1 LS-DYNA

*Users are strongly advised to run each LS-DYNA analysis in a separate directory. Some of the default names for the files generated by LS-DYNA that T/HIS can read are not unique and T/HIS can not tell which files belong to which model. If you do read multiple models from the same directory T/HIS will generate a warning message if you read the same file for more than 1 model.*

#### 5.1.1.1 Selecting Models

There are three ways to select the LS-DYNA models that you want to read into T/HIS

- (i) Select a single model (see [Section 5.1.1.1.1](#))
- (ii) Search directories for results and open open multiple models (see [Section 5.1.1.1.2](#))
- (iii) Open a model database and select the models you want to read ( see [Section 5.1.1.1.3](#))



### 5.1.1.1.1 Select Model

Select ANY results file from a model. T/HIS will then search for all the results files in that directory produced by the same analysis as the selected file (as illustrated on the right) and display a list of all the files found. The user can then select which files to open. The default is to open all the available results files.

If you are using the Oasys Ltd. SHELL to submit jobs then the default filenames will be "jobname.thf", "jobname.xtf", "binout", "abstat" etc. If you use the standard LSTC output file names then the filenames will be "d3thdt", "xtfile", "binout", "abstat".

The T/HIS preference option "this\*file\_names" can be used to set the default filenames that T/HIS searches for to either the ARUP set or the LSTC names.

When the user selects **Apply** the selected file are then opened and the contents scanned. After the files have been scanned the list of available data types will automatically be displayed ([see Section 5.1.1.5](#))

Read Data			
LS-DYNA	Groups	Keyword	T/HIS Curve
Bulk Data	Keyboard	CSV	Screen
ISO	LS-PREPOST	DIAdem	NASTRAN
Open : Single Model			
Filename : E:\test\sled\new_lg09.ptf			
<input type="checkbox"/> Extract curves to match model : 1			
<input type="checkbox"/> Overwrite existing curves			
<input checked="" type="checkbox"/> Copy curve styles		<input type="checkbox"/> Use default styles	
<input type="checkbox"/> Set styles		Colour	Width
		Style	Symbol
		Copy ▶	Copy ▶
		Copy ▶	Copy ▶
Model Unit System : Undefined			
<b>Apply</b>			
THF/d3thdt File			
<input checked="" type="checkbox"/> E:\test\sled\new_lg09.thf			
XTF/xtfile File			
<input checked="" type="checkbox"/> E:\test\sled\new_lg09.xtf			
LSDA/binout Database			
<input checked="" type="checkbox"/> E:\test\sled\binout			
ASCII Files			
<input checked="" type="checkbox"/> deforc		glstat	
matsum		nodout	
rcforc		sbtout	
sleout		spcforc	
ZTF - Additional Model data			
<input checked="" type="checkbox"/> E:\test\sled\new_lg09.ztf			

### 5.1.1.1.2 Search Directories Recursively

Multiple models can be opened by using the option to search directories recursively.

After a directory has been specified T/HIS will display a list of all the models it can find in the directory structure and each file can be selected

Read Data

LS-DYNA	Groups	Keyword	T/HIS Curve
Bulk Data	Keyboard	CSV	Screen
ISO	LS-PREPOST	DIAdem	NASTRAN

Open : Multiple Models (Search Directori
Directory : e:\test\crush

☐ Extract curves to match model : 1
☐ Overwrite existing curves
☒ Copy curve styles
☐ Use default styles
☐ Set styles

Colour

Width

Style

Symbol

Copy

Copy

Copy

Copy

Model Unit System : Undefined

Apply

☒ THF/d3thdt File
☒ XTF/xtfile File
☒ LSDA/binout Database
☒ ASCII Files

☒ ☒ All Models

☒ ..\BASE
☒ ..\RUN1
☒ ..\RUN2
☒ ..\RUN3
☒ ..\RUN4



### 5.1.1.1.3 Select Models From Database

From version 10.0 onwards T/HIS can select models from a model database. The database file is an XML format file that contains information on where models are located along with a brief description of each model, (see [below](#) for more details on the file format)

To select a model database either enter it's name in the text box or use the file selector.

The default model database can be specified as a command line argument (see [section 1.3](#) for more details). The default database filename and location can also be specified in the preference file (see [Appendix H](#) for more details)

```
this*database_dir:
this*database_file:
```

After a database file has been selected it's contents will be read and T/HIS will display a Tree Like menu showing the contents of the database.

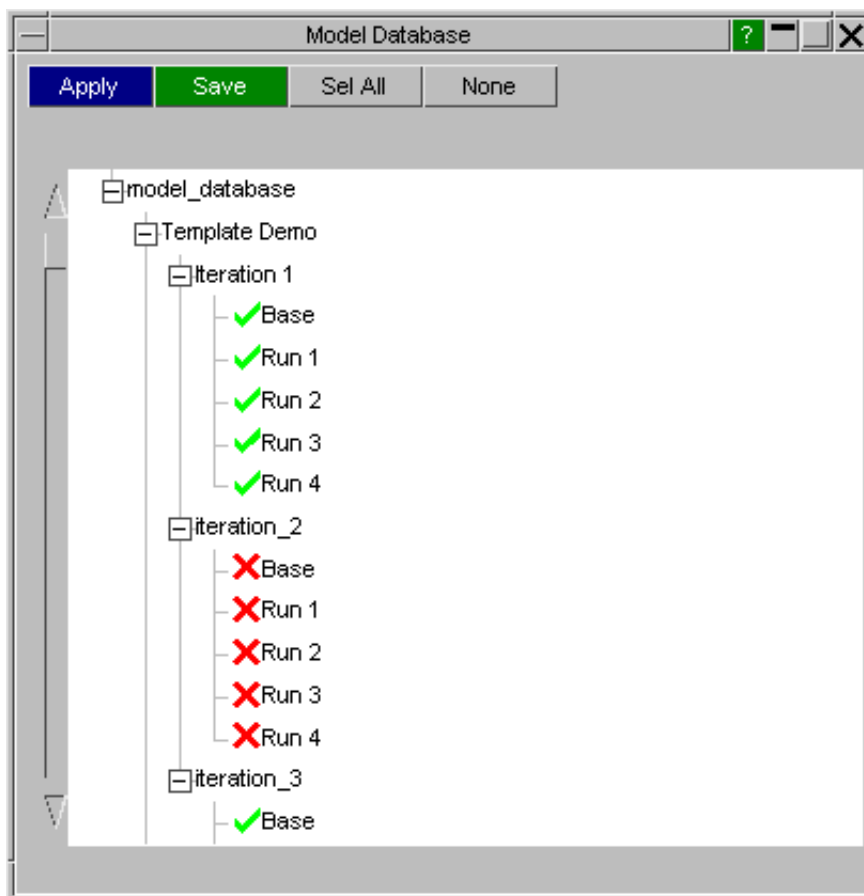
As each item is displayed T/HIS will check to see if the files that it refers to exist.

If a file does exist then a green ✓ tick will be displayed

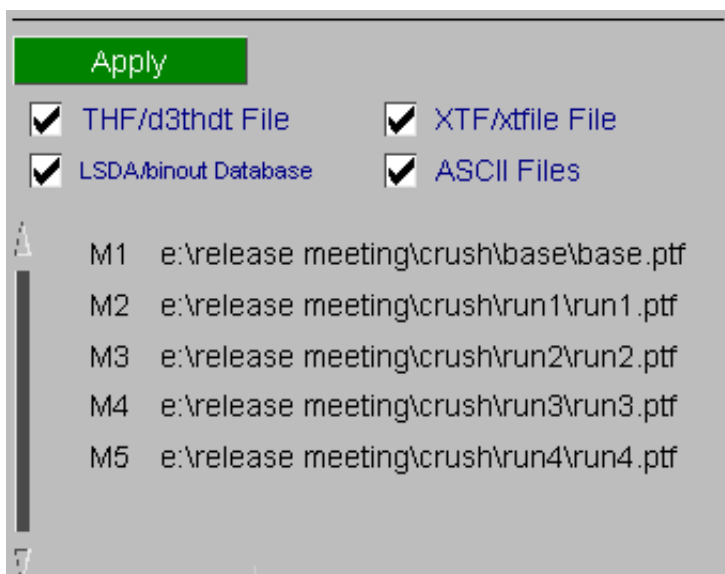
If a file does not exist then a red ✗ cross will be displayed

The number of levels in the database that are automatically expanded when it is first displayed can be specified in the preference file (see [Appendix H](#) for more details)

this\*database\_expand:



After selecting the required models use **Apply** to close the database window and return to the main menu where the selected models will be displayed along with the model numbers they will be read in as.

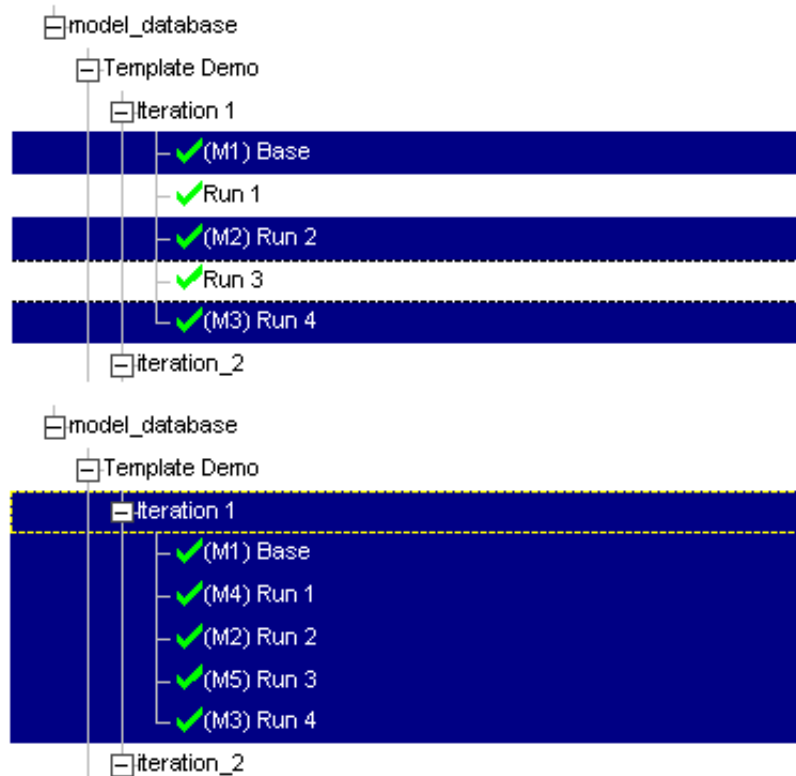


## Selecting Models

Models can be selected and deselected by clicking on each row. Multiple model can be selected by clicking on the 1st model and holding down SHIFT while selecting the last model in the range.

As each model is selected the model number than it will be read in as is automatically displayed alongside the model description.

A complete branch can be selected/deselected by selecting the branch label (Iteration 1).



## Modifying the Database

Database entries can be added, removed and modified by right clicking on a branch label or a model description

Right clicking on a branch label will display 4 options

**Modify**      Modify the branch label.

...

**Add Model**      Add a new model into the selected branch.

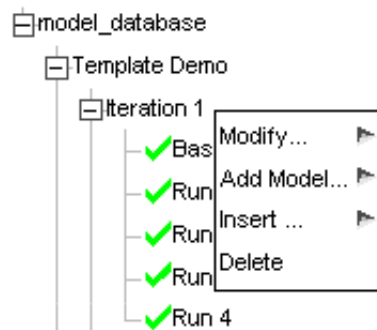
...

A menu will be displayed to select a new model and to define the model description that is displayed for the new model.

**Insert**      Insert a new branch within the selected branch.

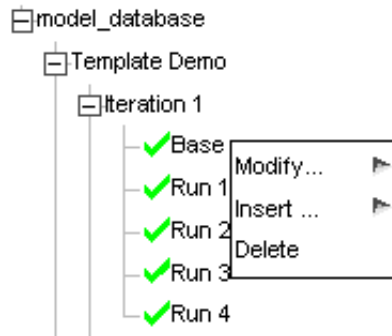
...

**Delete**      Delete this branch and everything within it.



Right clicking on a model description will display 3 options

- Modify** ... Modify the model location and description.
- Insert** ... Insert a new branch.  
The selected model will be moved into the new branch.
- Delete** Delete the model



## Saving the Database



After modifying the database use the **Save** option to save the changes for future sessions.



## Creating a new Database

If you do not have a database or if you want to create a new one then T/HIS can create the new database for you. To create a new database click the **CREATE** button and simply enter the name of the new database file in the text box that appears, T/HIS will then check that the file does not already exist and if it doesn't it will create a new empty database.

Alternatively if you type in the name of a file in the main Open Plot File window that does not exist then T/HIS will ask if you want to create a new empty database using that filename.

Once you have done this you can use the Modify options above to add items into the database and then save the file before exiting.

## Database Format

The Model Database uses an ASCII XML file format.

All items with the database are either branches or models. Each database entry has an XML name and a LABEL element. Models also contain a model element that contains the full pathname of one of the files belonging to the model.

The XML name should be unique and should obey the following rules

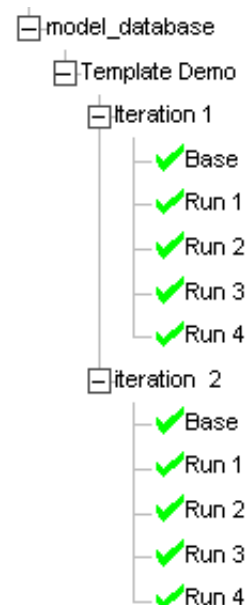
- Names can contain letters, numbers, and other characters
- Names must not start with a number or punctuation character
- Names must not start with the letters xml (or XML, or Xml, etc)
- Names cannot contain space

The LABEL is the string used to display an item within the tree view. Unlike the XML name the LABEL can contain any ASCII character.

```

<model_database version="10.000000">
  <Template_Demo label="Template Demo">
    <iteration_1 label="Iteration 1">
      <base label="Base"
        model="e:\release
meeting\crush\base\base.ptf"/>
      <run_1 label="Run 1"
        model="e:\release
meeting\crush\run1\run1.ptf"/>
      <run_2 label="Run 2"
        model="e:\release
meeting\crush\run2\run2.ptf"/>
      <run_3 label="Run 3"
        model="e:\release
meeting\crush\run3\run3.ptf"/>
      <run_4 label="Run 4"
        model="e:\release
meeting\crush\run4\run4.ptf"/>
    </iteration_1>
    <iteration_2 label="Iteration 2">
      <base label="Base"
        model="e:\test\crush2\base\base.ptf"/>
      <run_1 label="Run 1"
        model="e:\test\crush2\run1\run1.ptf"/>
      <run_2 label="Run 2"
        model="e:\test\crush2\run2\run2.ptf"/>
      <run_3 label="Run 3"
        model="e:\test\crush2\run3\run3.ptf"/>
      <run_4 label="Run 4"
        model="e:\test\crush2\run4\run4.ptf"/>
    </iteration_2>
  </Template_Demo>
</model_database>

```



### 5.1.1.2 Automatic extraction of model results

When a second or subsequent model is opened in T/HIS this option can be used to automatically generate the same curves as those already read from another model.

This option can also be used if a model is re-read into T/HIS to extract the same curves as those that had already been read from the model.

By default this option will attempt to generate curves that match those already read from model 1. If results have already been read from more than one model then the model to match the curves form can be set to any of the existing models.

This option can be used to overwrite the existing curves from a model. If a model has been read into T/HIS and curves have been read from the model while the analysis was still running then this option can be used to automatically update the curves.

<input checked="" type="checkbox"/>	Extract curves to match model :	1								
<input type="checkbox"/>	Overwrite existing curves									
<input checked="" type="checkbox"/>	Copy curve styles	<input type="checkbox"/> Use default styles								
<input type="checkbox"/>	Set styles	<table border="1"> <thead> <tr> <th>Colour</th> <th>Width</th> <th>Style</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>Copy ▶</td> <td>Copy ▶</td> <td>Copy ▶</td> <td>Copy ▶</td> </tr> </tbody> </table>	Colour	Width	Style	Symbol	Copy ▶	Copy ▶	Copy ▶	Copy ▶
Colour	Width	Style	Symbol							
Copy ▶	Copy ▶	Copy ▶	Copy ▶							

☒ Overwrite existing curves

When the curves from the 2nd or subsequent model are automatically generated then by default they will be given the same colours, and line styles as the curves in the original model.

Instead of copying the curve styles a new style for all the automatically generated curves can be specified. This make it very easy to set the same style for all of the curves that are read from a model. Alternatively the default T/HIS curve styles can be used.

<input checked="" type="checkbox"/> Copy curve styles	<input type="checkbox"/> Use default styles			
<input type="checkbox"/> Set styles	Colour	Width	Style	Symbol
	Copy ▶	Copy ▶	Copy ▶	Copy ▶

### 5.1.1.3 Model Unit System

This option can be used to set the default Unit System that will be applied to the model. For more information on Units see [Section 5.22](#)

Model Unit System :	Undefined ▼
---------------------	-------------

### 5.1.1.4 Entity Types

Items are shown in bright green if they occur in all the models that have been read into T/HIS and are currently selected. If they occur in at least one model but not all models then they are shown in a duller green (in the case shown in the adjacent picture Beams and Joints can be found in some but not all of the models).

Read Data ? X			
LS-DYNA	Groups	Keyword	T/HIS Curve
Bulk Data	Keyboard	CSV	Screen
ISO	LS-PREPOST	DIAdem	NASTRAN
Global	Parts	Part Groups	Nodes
Solids	Beams	Shells	Tk Shells
Stonewalls	Springs	Airbags	Contacts
Geo Contacts	Seatbelts	Retractors	Sliprings
Reactions	Joints	X Sections	Subsystems
Rigid Bodies	Spotwelds	SPCs	Boundarys
FSIs	SPHs	TRACERs	

### 5.1.1.5 Data Components

When reading data from any of the LS-DYNA binary files or the LS-DYNA ASCII files multiple components and entities may be selected at the same time.

Each data extraction menu consists of a list of available data components and a list of entities.

#### Data Components

Individual data components can be selected using the mouse. If a component has been selected and a second item is subsequently selected the first item will be deselected.

Multiple components may be selected by

1. Holding down the **CTRL** key when selecting items to add individual items to the list of selected components.
2. Holding down the **SHIFT** key when selecting items to add a range of items to the list of selected components.
3. Clicking on the first item to be selected and then dragging down the list of items without letting go of the mouse button.

Select Models    New Model    Reread Model

Model Title :

Output curve: # (first free)  ...

Key in :

Apply

Data Components

- KE - Kinetic energy
- IE - Internal energy
- HG - Hourglass energy
- TE - Total energy
- XM - X momentum
- YM - Y momentum
- ZM - Z momentum
- VX - Average X velocity
- VY - Average Y velocity
- VZ - Average Z velocity
- MA - Added mass

Entities

Sort By Model ▼

All    None

Select PART(s)

- M1:Part 1 : (UPPER
- M1:Part 54 : (UPPE
- M1:Part 69 : (RIDED
- M1:Part 70 : (BREA
- M1:Part 1000 : (WIN
- M1:Part 1001 : (FLO
- M1:Part 1002 : (TOE
- M1:Part 1003 : (SEA
- M1:Part 1006 : (SEA
- M1:Part 1110 : (KNE
- M1:Part 1136 : (RIGI
- M1:Part 1137 : (RIGI
- M1:Part 1138 : (KNE

### 5.1.1.6 Entities

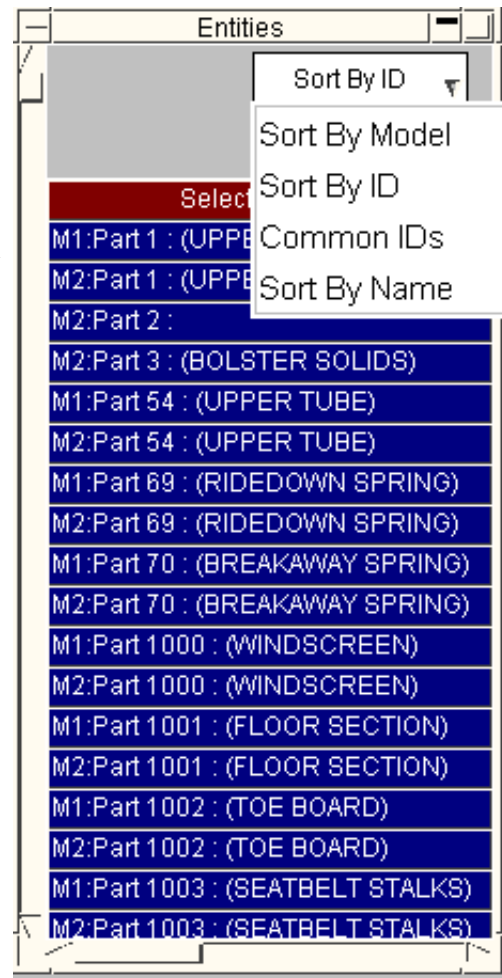
Individual entities can be selected/deselected using the mouse.

Multiple entities may be selected by

1. Holding down the **CTRL** key when selecting items to add them to the list of selected entities.
2. Holding down the **SHIFT** key when selecting items to add a range of items to the list of selected entities.
3. Clicking on the first item to be selected and then dragging down the list of items without letting go of the mouse button.

Entities can be sorted in four ways:

<b>Sort by model</b>	will list all entities in the lowest number model in order of ascending ID number, then all entities in the next-lowest model, and then move through the rest of the models in ascending order.
<b>Sort by ID</b>	will list all entities in ascending order showing the model ID for each entity
<b>Common IDs</b>	will list only the entities with IDs that are common to all models without showing the model ID's
<b>Sort by Name</b>	arranges the entities in alphabetical order based on their names.





### 5.1.1.7 Surfaces/Integration Points

Some BEAM, SHELL, and THICKSHELL data components can be read from multiple integration points.

If a data component is available for multiple integration points then an additional **Select Surface** options is displayed.

The screenshot shows the 'Select Models' dialog box. At the top are three buttons: 'Select Models' (green), 'New Model' (green), and 'Reread Model' (green). Below them are input fields for 'Output curve: % (highest+1)' and 'Key in :'. To the right is an 'Apply' button. A '<.. Go Back' button is at the top left of the main content area. Below it is a 'Select Surfaces' button. A list of stress and strain components is shown: SXX - Stress in XX, SYY - Stress in YY, SZZ - Stress in ZZ, SXY - Stress in XY, SYZ - Stress in YZ, SZX - Stress in ZX, SMX - MAX principal stress, SMN - MIN principal stress, SMS - MAX shear stress, SVM - von Mises stress, SAV - Average stress (Pressure), and EPS - Effective plastic strain. To the right of this list is a 'Sort By Model' dropdown menu and two buttons: 'All' (red) and 'None' (green). Below these is a 'Select Surface(s)' section with a list of surfaces: M1:Shell 19525, M1:Shell 19528, M1:Shell 19535, and M1:Shell 19538.

#### Select Surface

This option will display a separate menu listing all of the integration points that are available to read data from.

For Shell and Thick Shell elements the menu will include all of the through thickness integration points plus 3 additional options; TOP, MIDDLE and BOTTOM.

For Beam elements the menu will just display the integration points.

#### In plane int points

In addition to the through thickness integration points recent versions of LS-DYNA can also output data for multiple in-plane integration points for fully integrated Shell and Thick Shell elements. If T/HIS can identify that the model contains data for multiple in-plane integration points then these options can be used to select the individual in-plane integration points or to average the 4 in-plane points.

For more information on selecting integration points for beams, shells and thick shells see Sections [A.6](#), [A.7](#) and [A.8](#)

The screenshot shows the 'Select Models' dialog box with the 'Surface / Int' section selected. It has the same top buttons and input fields as the previous screenshot. Below the '<.. Go Back' button is a section titled 'In plane int points' with four checkboxes: 'Average', 'In plane #1' (checked), 'In plane #2', 'In plane #3', and 'In plane #4'. To the right is a 'Surface / Int' dropdown menu and two buttons: 'All' (red) and 'None' (green). Below these is a list of integration points: Top, Middle, Bottom, Int Point #1, Int Point #2, Int Point #3, Int Point #4, Int Point #5, Int Point #6, Int Point #7, and Int Point #8.

### 5.1.1.8 Shell and ThickShell Data Components

If Shell and ThickShell data is being read from the LSDA (binout) file then the file can contain data components in both the ELOUT and ELOUTDET branches.

By default T/HIS uses the data from ELOUTDET as ELOUT only contains a subset of the data in ELOUTDET.

In some versions of LS-DYNA it is possible to change the Shell and ThickShell data components written to the ELOUT so that they are defined using the global coordinate system (see EOCS on \*CONTROL\_OUTPUT) instead of the default element local coordinate system. If this option is used then only the ELOUT file is modified, the ELOUTDET file is still written using the local coordinate system.

#### Use ELOUT instead of ELOUTDET

If T/HIS detects that the LSDA file contains both ELOUT and ELOUTDET and that they are using different coordinate systems then this option can be used to force T/HIS to use the ELOUT file data components using the global coordinate system.

This option can also be set via the preference file (see [Appendix H](#) for more details) and via the command line (see [Section 1.3](#))

Global	Parts	Part Groups	Nodes
Solids	Beams	<b>Shells</b>	Tk Shells
Stonewalls	Springs	Airbags	Contacts
Geo Contacts	Seatbelts	Retractors	Sliprings
Reactions	Joints	X Sections	Subsystems
Rigid Bodies	Spotwelds	SPCs	Boundarys
FSIs	SPHs	Tracers	Pulleys

Select Models	New Model	Reread Model
Output curve: % (highest+1)		...
Key in :		Apply

STRESS Tensor components
PLASTIC STRAIN
STRAIN Tensor components
FORCE/MOMENT components
MISCELLANEOUS components
EXTRA components

  
☒ Use ELOUT instead of ELOUTDET

The LSDA (binout) file contains ELOUT and ELOUTDET data. The ELOUT file uses the Global coordinate system for Shell and ThickShell results while ELOUTDET is using a Local coordinate system.

## 5.1.2 GROUPS

This option can be used to read a file containing PART group definitions. If a model is read in which contains PART information then the PART groups can be used to read in and sum energies for a group of PARTS in one go.

The 1st time T/HIS finds a group file ( groupXXX.asc ) in a directory it will automatically read the file and create the PART groups.

After reading the 1st group file T/HIS will by default ignore any other group files it finds in directories when it opens a model.

This option can be changed as follows.

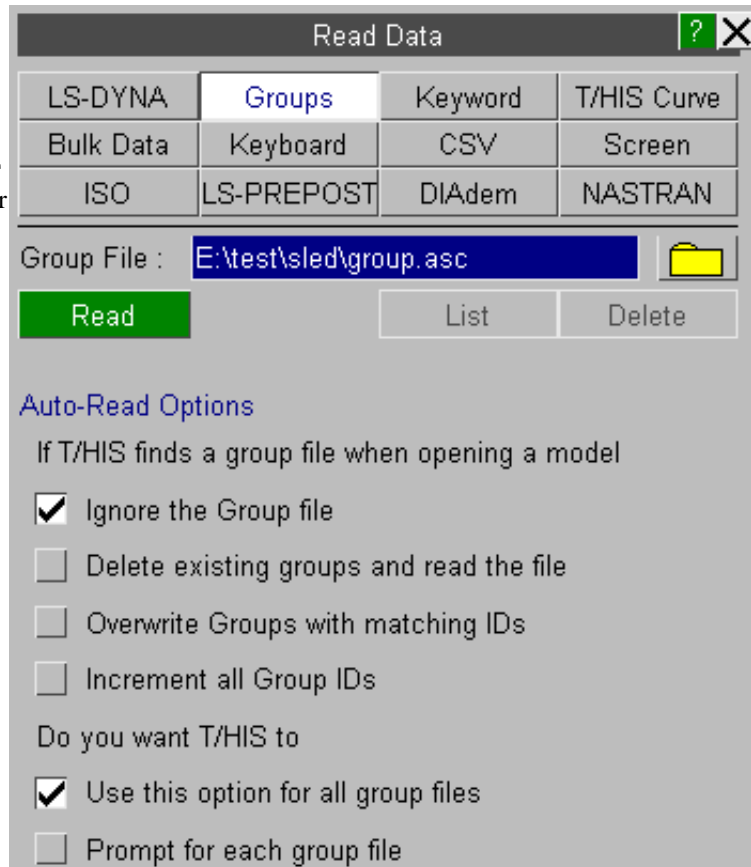
- |                  |  |
|------------------|--|
| <b>Ignore</b>    | This option (the default) will make T/HIS ignore any more group files it finds   |
| <b>Delete</b>    | If T/HIS finds a group file when a new model is read in then all existing group definitions will be deleted before the new file is read  |
| <b>Overwrite</b> | If T/HIS finds a group file when a new model is read in then all the new group definitions will be added to the existing ones. If the new file contains a group with the same ID as an existing group then the old definition will be overwritten. |
| <b>Increment</b> | If T/HIS finds a group file when a new model is read in then all the new group definitions will be added to the existing ones but the group ID's will be incremented to ensure that they do not clash with existing ones.                          |

The default option can be changed using the preference option

this\*read\_group\_files:

(see [Appendix H](#) for more details)

If the option to read groups files is set and the directory contains more than one group file then T/HIS will use the newest file.



### 5.1.3 T/HIS Curve

This option can be used to read in curves stored in T/HIS curve file format (see [Appendix B](#) for more details)

By default this option can be used to select a single file. After selecting the file it will automatically be opened and read and all of the curves in the file will be read in.

Read Data

LS-DYNA	Groups	Keyword	T/HIS Curve
Bulk Data	Keyboard	CSV	Screen
ISO	LS-PREPOST	DIAdem	NASTRAN

☒ Curve File :

☐ Search Directories Recursively :

Output Curve :  ...

☒ Read Style Data

☐ Filter :

In addition to reading a single file this option can also be used to search directories recursively for multiple files.

After the search has finished a list showing all of the files that have been found will be displayed so that multiple files can be selected and read in one operation.

By default T/HIS will search for files with the file extension .cur, this can be changed if required.

In addition to changing the default file extension the list of files can also be filtered. The filter string can contain the following wildcards

- \* matches multiple characters
- ? matches a single character

**Note : The filtering ignores case.**

☐ Curve File :

☒ Search Directories Recursively :

Output Curve :  ...

☒ Read Style Data

☒ Filter :

☒ ☒ All Curve Files

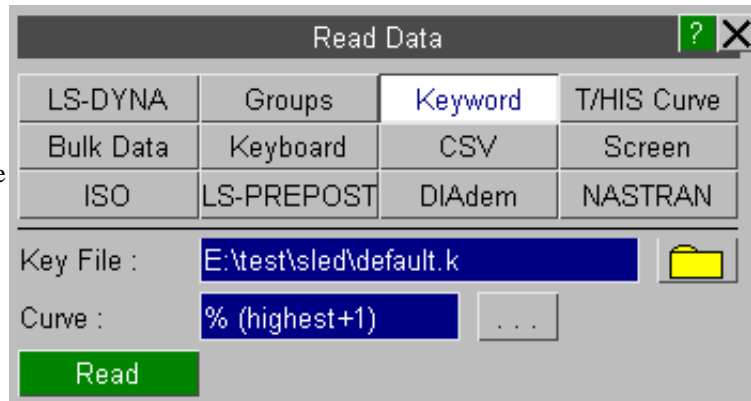
- ☒ ..\CRUSH\BASE\base.cur
- ☒ ..\CRUSH\BASE\results.cur
- ☒ ..\CRUSH\RUN1\results.cur
- ☒ ..\CRUSH\RUN1\run1.cur
- ☒ ..\CRUSH\RUN2\results.cur
- ☒ ..\CRUSH\RUN2\run2.cur
- ☒ ..\CRUSH\RUN3\results.cur
- ☒ ..\CRUSH\RUN3\run3.cur
- ☒ ..\CRUSH\RUN4\results.cur

### 5.1.4 KEYWORD

Read data into T/HIS from an LS-DYNA KEYWORD input file. All X,/Y data defined using **\*DEFINE CURVE** will be read in from the specified input file. Any X and Y axis scaling or offsets defined within the **\*DEFINE CURVE** definition will be applied to the X,Y as it is read in. If the **TITLE** option has been used the title will be used as the curve label otherwise the curve ID number will be used.

From version 9.3 onwards this option will also process any files specified using the **\*INCLUDE** option.

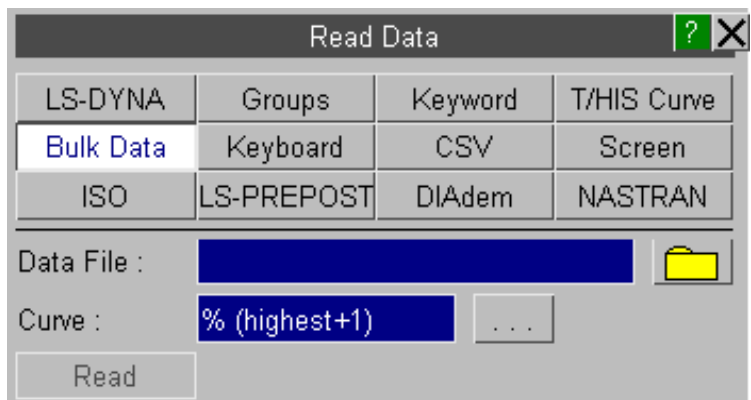
```
/re kw      read all curves from
"filename"  KEYWORD input file
            "filename"
```



### 5.1.5 BULK

Read data into T/HIS from a Bulk Data file. The format of a Bulk Data file is described in [Appendix C](#).

```
/re bd      read all curves from Bulk
"filename"  Data file "filename"
```



### 5.1.6 KEYBOARD

Key in curve information directly. A dialogue window is displayed upon requesting this option where the user will be prompted for title, x and y axis labels, a curve identifier and then a series of points. Once all the points required have been entered carriage return should be pressed. The user will then be prompted for the curve or file in which to store this data : # means use the next free curve.

## 5.1.7 CSV

The **CSV** menu (see right) can be used to specify the name of a comma separated variable file to read into T/HIS.

The file may contain up to 1000 columns of data (separated by commas).

The maximum line length supported by this option is 10240 characters.

### File Format

This option can be used to change the CSV file format between the X,Y,X,Y,X,Y format where alternate columns are the X and Y values for each curve and the X,Y,Y,Y format where there is a single column containing the x-axis values for all the curves.

### Field Separator

By default T/HIS assumes that the columns of data are separated by commas, this option can be used to change the field separator to either a Tab or Spaces.

If the 'Space' option is used then multiple spaces are counted as a single field separator. If curve or axis labels are defined in the file and they contain spaces then they need to be enclosed in pairs of " quotes.

The default field separator can specified in the preference file (see [Appendix H](#) for more details)

```
this*csv_separator:
```

### Read X Values

This option can be used to specify a column within the file that contains the X-axis data values that should be used for all of the other columns of data.

### Generate X Values

This option can be used to automatically generate the X-axis values if none of the columns within the file contain the data.

### Read Labels

This option can be used to specify a row within the file that contains labels for each of the columns of data that can be used as the curve labels within T/HIS.

**Generate Labels**

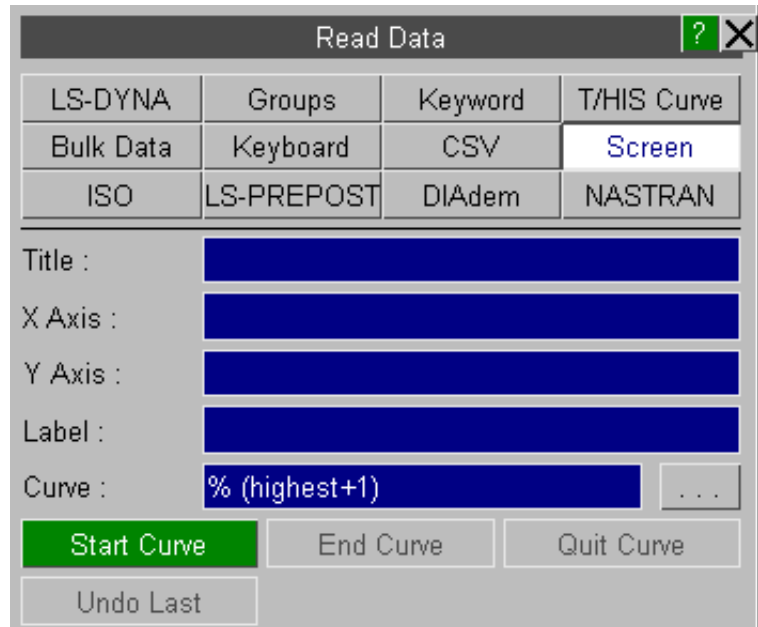
This option can be used to automatically generate labels for each set of data. A single string can be specified which will then have the column number appended to it to generate unique labels.

**Read Axis Labels**

This option can be used to specify a row within the file that contains the axis labels.

## 5.1.8 SCREEN

The **SCREEN** menu (see right) can be used to interactively create a curve T/HIS by selecting points using the mouse.

**Start Curve**

This option will start point selection process. Once you have started creating a curve all the other T/HIS menus will be disabled until you end the point selection using either the **End Curve** or **Quit Curve** options.

*Dynamic viewing will still be available.*

**End Curve**

This option will end the current curve creation and save the curve.

**Quit Curve**

This option will end the current curve creation without saving the curve.

**Undo Last**

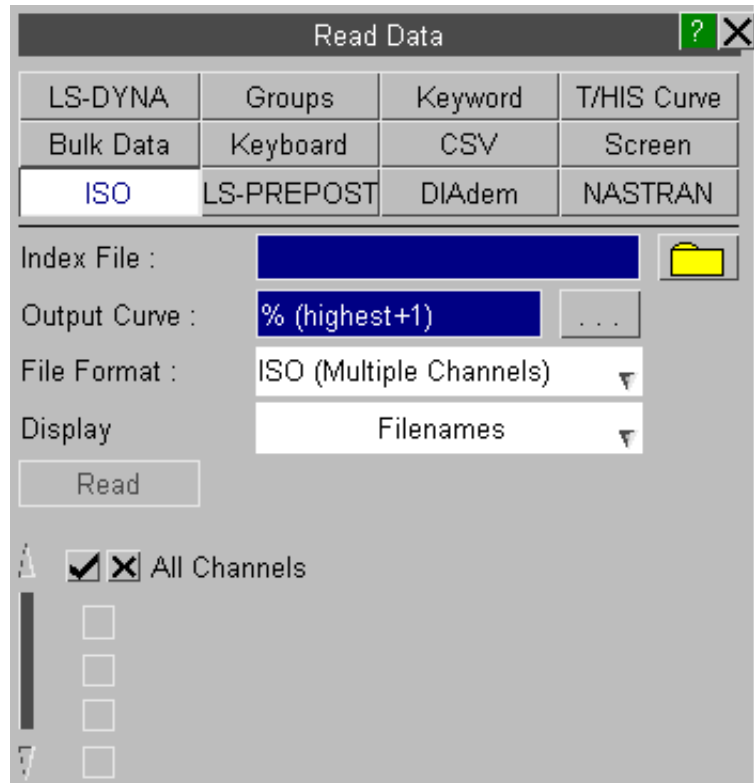
This option can be used delete the last point created (the middle mouse button will also delete the last point).

## 5.1.9 ISO

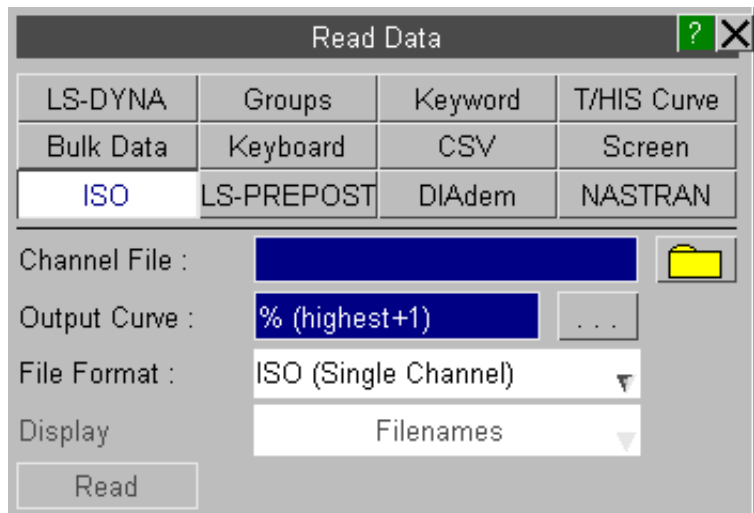
This option can be used to read in curves from files written using the ISO/TS 13499:2003 file format.

The default option in T/HIS is to read in an Index file containing information on multiple channels. After the file has been opened and read a list of all the available channels will be displayed so the required channels can be selected.

When listing the channels the default is to display the filenames for each of the channel files. Alternatively the channel names (read from the Index) file can be displayed.



Instead of reading an Index file and then selecting which channels to read individual channel file can be read in directly.





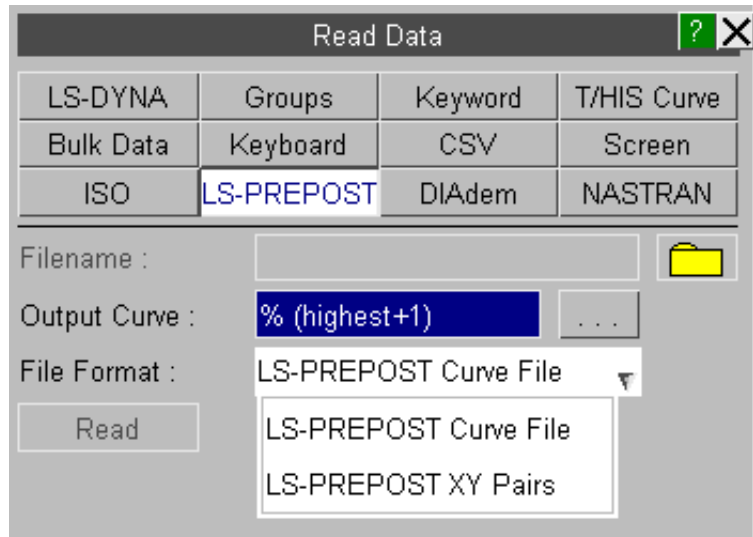
### 5.1.10 LS-PREPOST

This option can be used to read in curves from files written out from LS-PREPOST.

Two different file formats are supported

#### LS-PREPOST Curve Files

#### LS-PREPOST XY Pairs



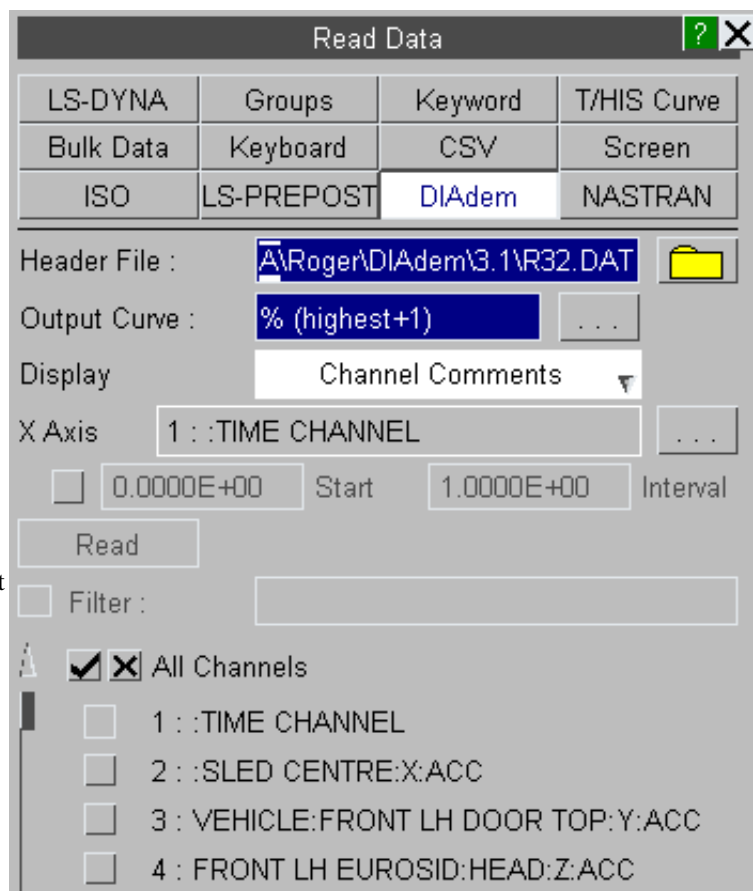
### 5.1.11 DIAdem

This option can be used to read in data from DIAdem format data files. After selecting a DIAdem header file a list of all the available channels will be displayed so the required channels can be selected.

Version 11.0 of T/HIS supports the following DIAdem data file formats

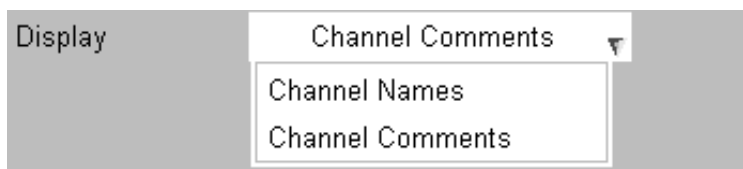
REAL32  
REAL48  
REAL64  
INT16  
INT32  
WORD8  
WORD32  
ASCII

The MSREAL32, TWOC12 and TWOC16 are not supported.



By default T/HIS will display the channel comments (header block 201) for each channel. This can be changed to the channel names (header block 200) using the popup menu if required.

When channels are read in this option is also used to create the labels for each curve.

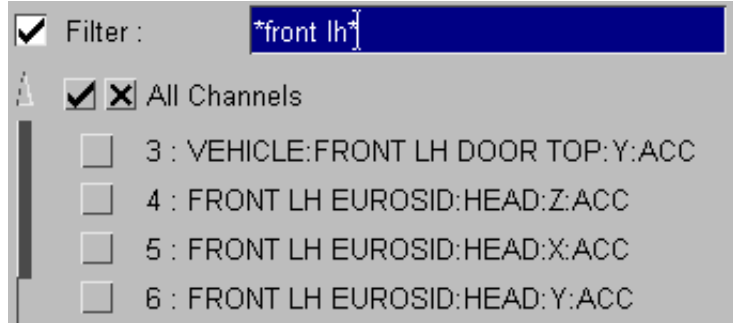


As well as displaying either the channel comments or the channels names the list of channels can also be filtered if required .

The filter string can contain the following wildcards

- \* matches multiple characters
- ? matches a single character

**Note : The filtering ignores case.**



Normally one of the DIAdem data channels contains the x-axis (time) values. By default T/HIS assumes this is channel 1 but this can be changed using the button labelled ...



If none of the channels contain the x-axis values then a start value and an increment can be specified to generate curves with evenly spaced x-axis values.

### 5.1.11.1 Supported DIAdem header file blocks

The following DIAdem header file data blocks are supported. All other data blocks are ignored.

#### GLOBAL HEADER

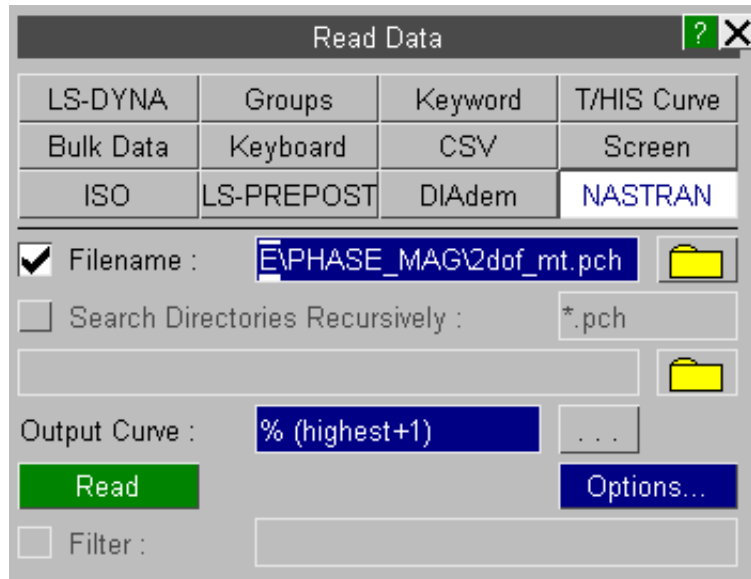
- 111 Value for NoValues in the data file
- 112 Interchange high- and low-bytes

#### CHANNEL HEADER

- 200 Channel name
- 201 Channel comment
- 210 Channel type
- 211 File from which channel data is read
- 213 Method of storing the data
- 214 Data type
- 220 No. of values in the channel
- 221 Pointer to the 1st value in the channel
- 222 Offset for ASCII block files
- Offset for binary block files with header
- 223 Local ASCII-pointer in the case of ASCII block files
- 230 Separator character for ASCII-block files
- 231 Decimal character in ASCII-files
- 232 Exponential character in ASCII-files
- 240 Exponential character in ASCII-files
- 241 Step width / Factor
- 252 Keyword for NoValues in the channel
- 254 Value for NoValues in the channel

## 5.1.12 NASTRAN

This option can be used to read in data from from NASTRAN PCH files.



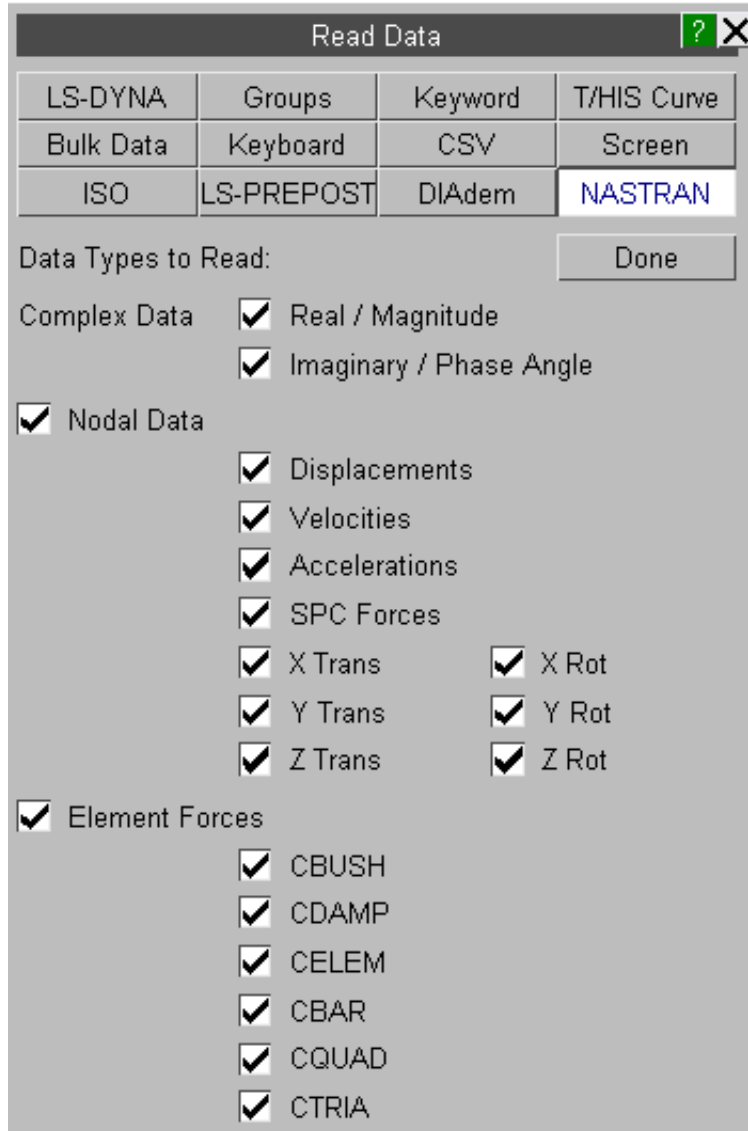
Currently the following types and data components are supported along with the SORT1, SORT2 and XYPUNCH file formats

Nodal	Displacements
Nodal	Velocities
Nodal	Accelerations
Nodal	SPC Forces
CBUSH	Element Forces
CDAMP	Element Forces
CELEM	Element Forces
CBAR	Element Forces
CQUAD	Element Forces
CTRL	Element Forces

By default T/HIS will read in every curve that is finds in the file so if you read in a file containing nodal displacements from a SORT2 format file you will end up with 12 curves being produced in T/HIS for each node.

X,Y,Z translation (Real) / (Magnitude)  
 X,Y,Z translation (Imaginary) / (Phase angle)  
 X,Y,Z rotational (Real) / (Magnitude)  
 X,Y,Z rotational (Imaginary) / (Phase angle)

The **Options...** button will display the following menu that will allow some components to be deselected before reading the file.



LS-DYNA	Groups	Keyword	T/HIS Curve
Bulk Data	Keyboard	CSV	Screen
ISO	LS-PREPOST	DIAdem	NASTRAN

Data Types to Read: Done

☒ Complex Data
 ☒ Real / Magnitude  
☒ Imaginary / Phase Angle  
☒ Nodal Data  
☒ Displacements  
☒ Velocities  
☒ Accelerations  
☒ SPC Forces  
☒ X Trans ☒ X Rot  
☒ Y Trans ☒ Y Rot  
☒ Z Trans ☒ Z Rot  
☒ Element Forces  
☒ CBUSH  
☒ CDAMP  
☒ CELEM  
☒ CBAR  
☒ CQUAD  
☒ CTRIA

**Complex Data** For complex data components written out as a pair of real and imaginary numbers or as a magnitude and phase angle either of the components can be deselected.

**Nodal Data** For nodal data any of the 4 data types can be deselected along with any of the 6 translational/rotational directions.

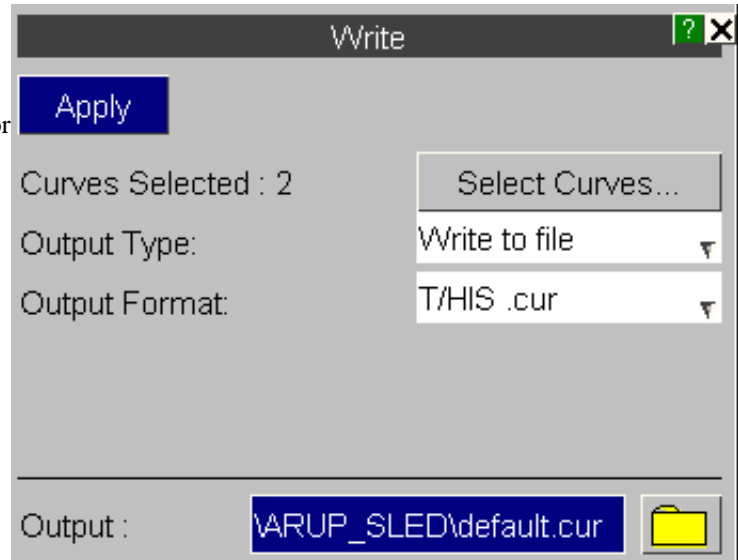
**Element Forces** For element forces each individual element type can be deselected.

T/HIS will automatically create curve labels for each curve generated from the PCH file. The entity types, ID's and components will also be stored with the curves to allow the curves to be sorted using the curve table (see [Section 5.3.4](#))

Curve Table								?				
Dismiss		View...	Update	Filter by :	Model...	Label...	Type...	Component...				
Select :	All	None	Clear All Filter Options									
ID	Label/Group Name	Directory	Model/File	Type	Entity ID	Component	Style	* 1				
1	Vel x - Node 1 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	1	Vel x	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
2	Vel y - Node 1 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	1	Vel y	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
3	Vel z - Node 1 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	1	Vel z	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
4	RVel x - Node 1 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	1	RVel x	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
5	RVel y - Node 1 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	1	RVel y	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
6	RVel z - Node 1 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	1	RVel z	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
7	Vel x - Node 2 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	2	Vel x	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
8	Vel y - Node 2 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	2	Vel y	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
9	Vel z - Node 2 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	2	Vel z	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
10	RVel x - Node 2 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	2	RVel x	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
11	RVel y - Node 2 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	2	RVel y	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
12	RVel z - Node 2 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	2	RVel z	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
13	Vel x - Node 3 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	3	Vel x	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
14	Vel y - Node 3 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	3	Vel y	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
15	Vel z - Node 3 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	3	Vel z	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
16	RVel x - Node 3 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	3	RVel x	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
17	RVel y - Node 3 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	3	RVel y	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
18	RVel z - Node 3 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	Node	3	RVel z	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
19	Force - CDAMP 5 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	CDAMP	5	Force	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
20	Force - CDAMP 6 : subcase 1	E:\test\PC\TIME	2dof_mt.pch	CDAMP	6	Force	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

## 5.2 WRITE Options

Writes a group of curves out to a file for later use or to the screen.

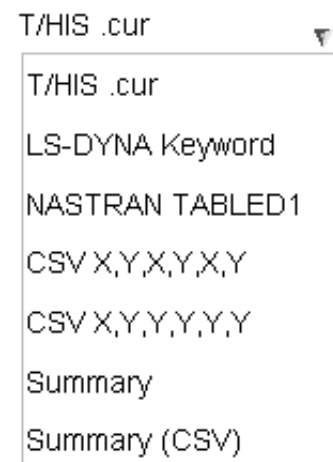


### 5.2.1 WRITE TO FILE

Writes a group of curves out to a file for later use if required. The user is prompted for the list of curves to write out after a filename has been specified.

#### 5.2.1.1 FILE FORMAT

Writes a group of curves out to a file for later use if required. The user is prompted for the list of curves to write out after a filename has been specified.



#### T/HIS .cur format

This option will write out curves using the default T/HIS curve format. One curve file will be written containing all the selected curves along with their Titles, Axis Labels, Line Labels and styles. From version 9.4 onwards the curve file can also contain information on the UNIT system and the X and Y axis units for each curve (see [Appendix B](#) for more details on the curve file format)

#### LS-DYNA Keyword

One file will be written containing all the selected curves using the LS-DYNA \*DEFINE\_CURVE format so that the file is suitable for inclusion in a LS-DYNA keyword file.

#### NASTRAN D1

This option will write out curves using the NASTRAN TABLE D1 format. Curves are listed sequentially in the file.

#### CSV X.Y.X.Y.X.Y

This option will write out curves using as a CSV (comma separated variable) file that can be read into other programs like Microsoft EXCEL. The columns written are x-values for the 1st selected curve, y-values for the 1st selected curve, x-values for the 2nd selected curve, y-values for the 2nd selected curve ...

**CSV  
X,Y,Y,Y,Y,Y**

This option also writes out a CSV file. All the curves are output using a single consistent set of X values that can either be taken from one of the curves or they can be generated automatically.

**Summary**

Gives a summary of the curve. This includes the type of data being plotted and the maximum and minimum values in the curve.

**Summary (CSV)**

CSV (comma separated variable) version of the summary file.

From version 9.4 onwards the CSV files generated by T/HIS can also contain information on the UNIT system and the X and Y axis units for each curve. If you don't want to output this information then you can turn it off.

The default setting for this option can be set via the preference option

**this\*write\_csv\_units:**

This option can also be turned on and off in FAST-TCF scripts (see section 7.XX)

**5.2.2 WRITE TO SCREEN**

Writes data to a text window on the screen.

**5.2.2.1 OUTPUT FORMAT**
**LIST**

This option will write out all the points in the selected curves.

**Summary**

Gives a summary of the curve. This includes the type of data being plotted and the maximum and minimum values in the curve.

**SCAN**




Scans a group of curves and reports the maxima and minima values for each individual curve along with the overall maxima and minima

## 5.3 Curve Manager

In screen menu mode curves are managed using the **CURVE MANAGER** window, shown in the figure (right).

By default the **CURVE MANAGER** menu only displays 1000 curves. An unlimited amount of curves can be used and these are displayed in the menu in blocks of 1000. If an attempt is made to use a curve higher than 1000 then the Range options are used to select which group of 1000 curves you wish to display.

Against each curve that currently contains information is a curve number button. The colour of this button indicates the current blanking status of a curve

	The curve is unblanked in all active graphs ( <a href="#">see section 3.5</a> )
	The curve is blanked in all active graphs
	The curve is unblanked in some active graphs

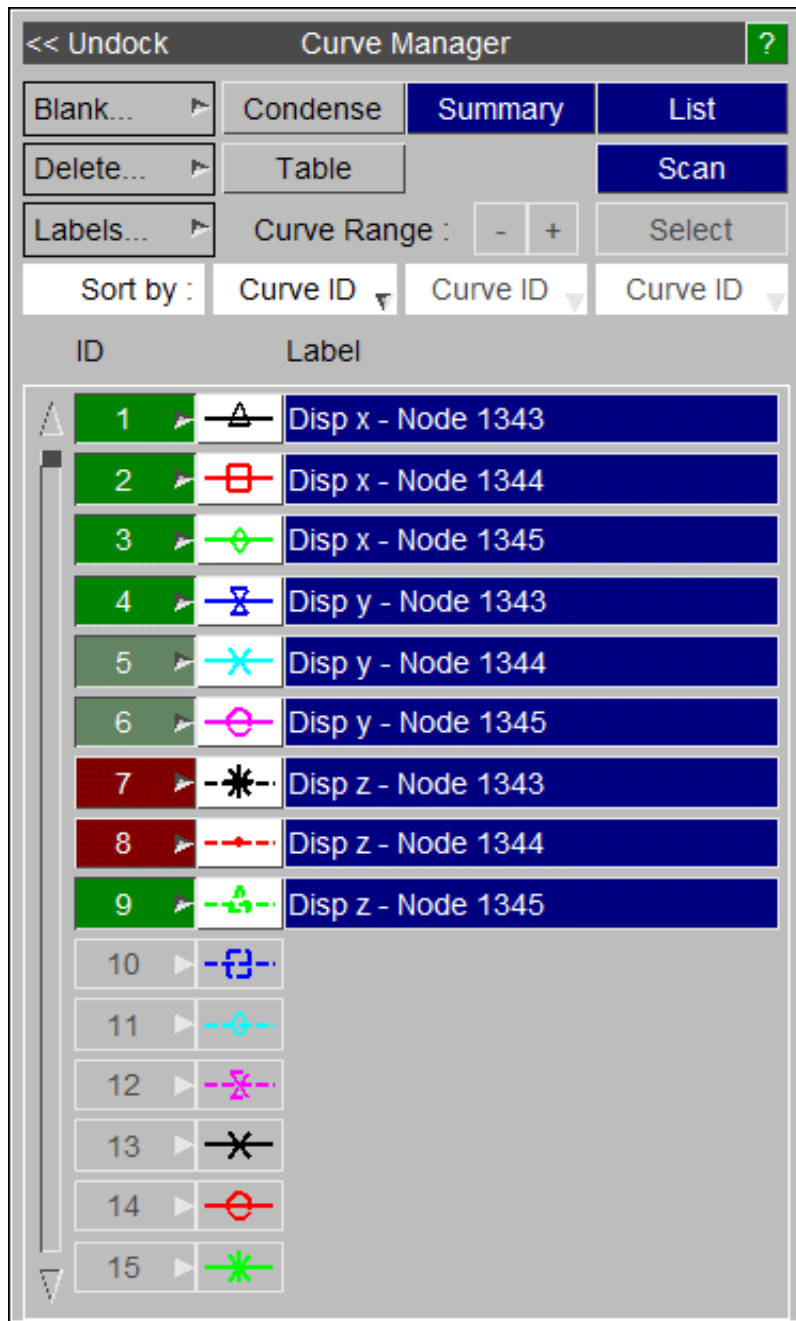
The blanking status of each curve can be changed by clicking on this button. The [Curve Table](#) can also be used to change the blanking status of a curve.

A range of curves may either be blanked or unblanked by selecting the first button in the range and then holding down the **SHIFT** key while selecting the last button in the range. All buttons that lie between the first and last buttons selected will have their status changed to match that of the first button that was selected.

The line label for each curve may be changed by over-typing the label currently displayed in the line label box.

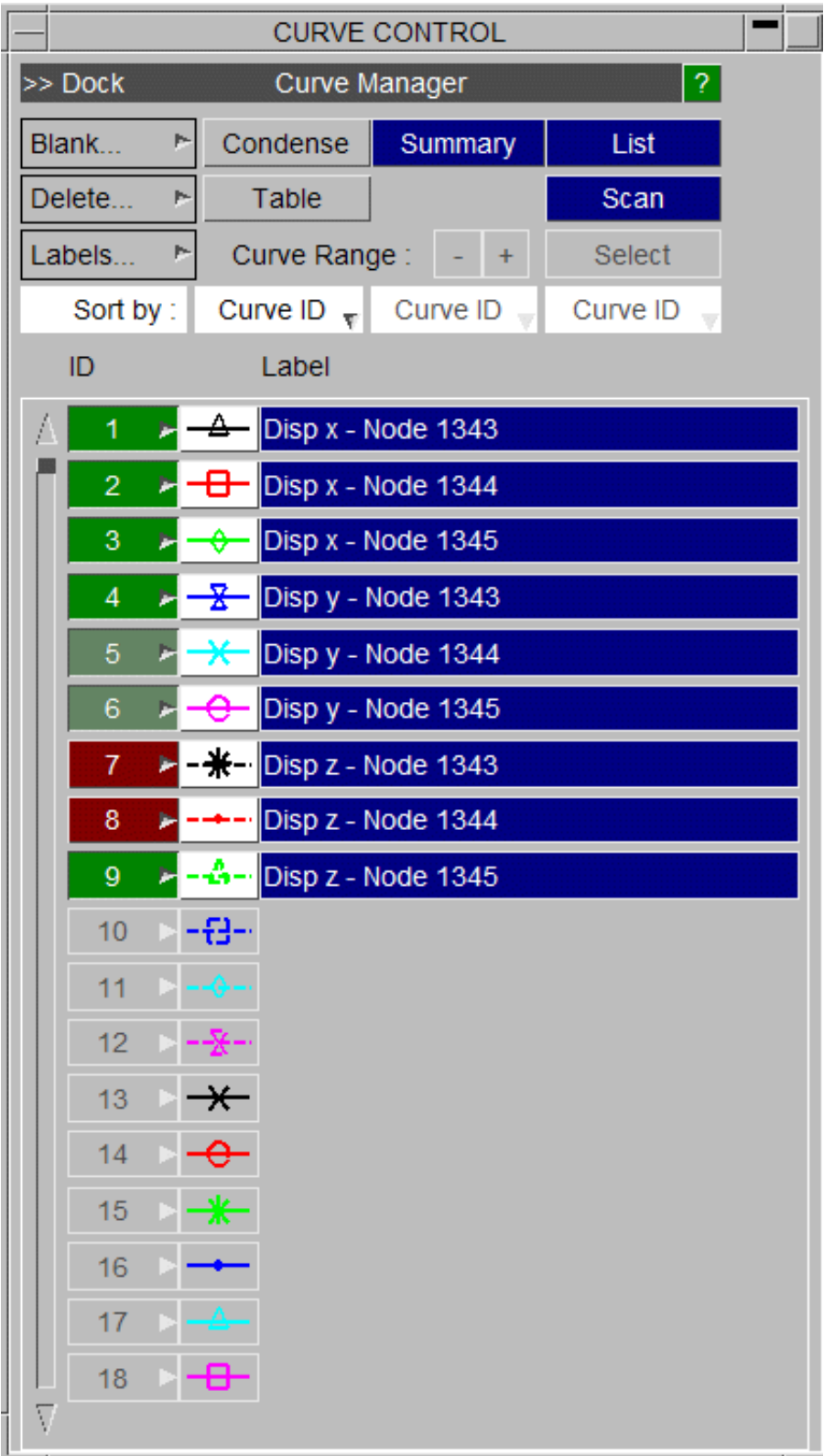
The button located between the curve number button and the curve label shows the current colour, line style and symbol that will be used to plot the curve. These properties can be modified by clicking on this button to display the line style menu, see [Section 5.6](#).

The **CURVE CONTROL** window can also be accessed via the **File....Curves** option at the top of the graphics window or from the **Curves** button in the main menu.



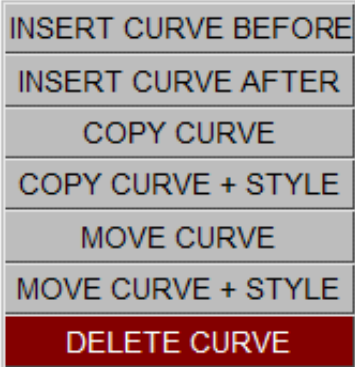


If the curve labels are too long to be seen in the standard Curve Manager menu then the menu can be turned into a floating menu by selecting the <<<Undock option in the menu header. After undocking the menu it can be re-docked by selecting >>>Dock.



### 5.3.1 Reordering Curves

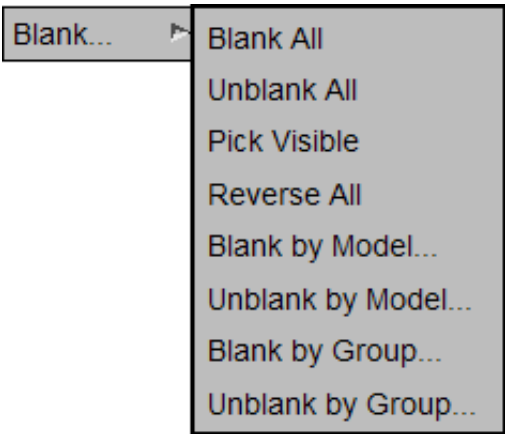
Attached to each of the curve number buttons is a popup menu that can be used to reorder curves by copying and moving them. This menu is accessed by clicking the right mouse button over the curve number buttons.



- INSERT CURVE BEFORE
- INSERT CURVE AFTER
- COPY CURVE
- COPY CURVE + STYLE
- MOVE CURVE
- MOVE CURVE + STYLE
- DELETE CURVE

- Inserts the last curve copies to a scratch definition before the selected curve.
- Inserts the last curve copies to a scratch definition after the selected curve.
- Copies the curve to a scratch definition.
- Copies the curve along with its line style settings to a scratch definition.
- Copies the curve to a scratch definition and then deletes the original curve
- Copies the curve along with its line style settings to a scratch definition and then deletes the original curve
- Deletes the selected curve

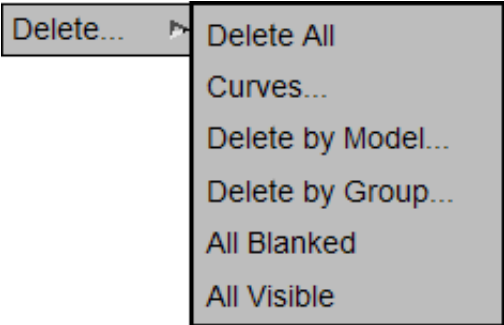
### 5.3.2 Blank...



- Blank All
- Unblank All
- Pick Visible
- Reverse All
- Blank by Model...
- Unblank by Model...
- Blank by Group...
- Unblank by Group...

- Blank all curves
- Unblank all curves
- Pick curves from the screen to be blanked.
- Reverse the blanking status of all curves
- Blank curves belonging to a Model
- Unblank curves belonging to a Model
- Blank curves by Curve Group
- Unblank curves by Curve Group

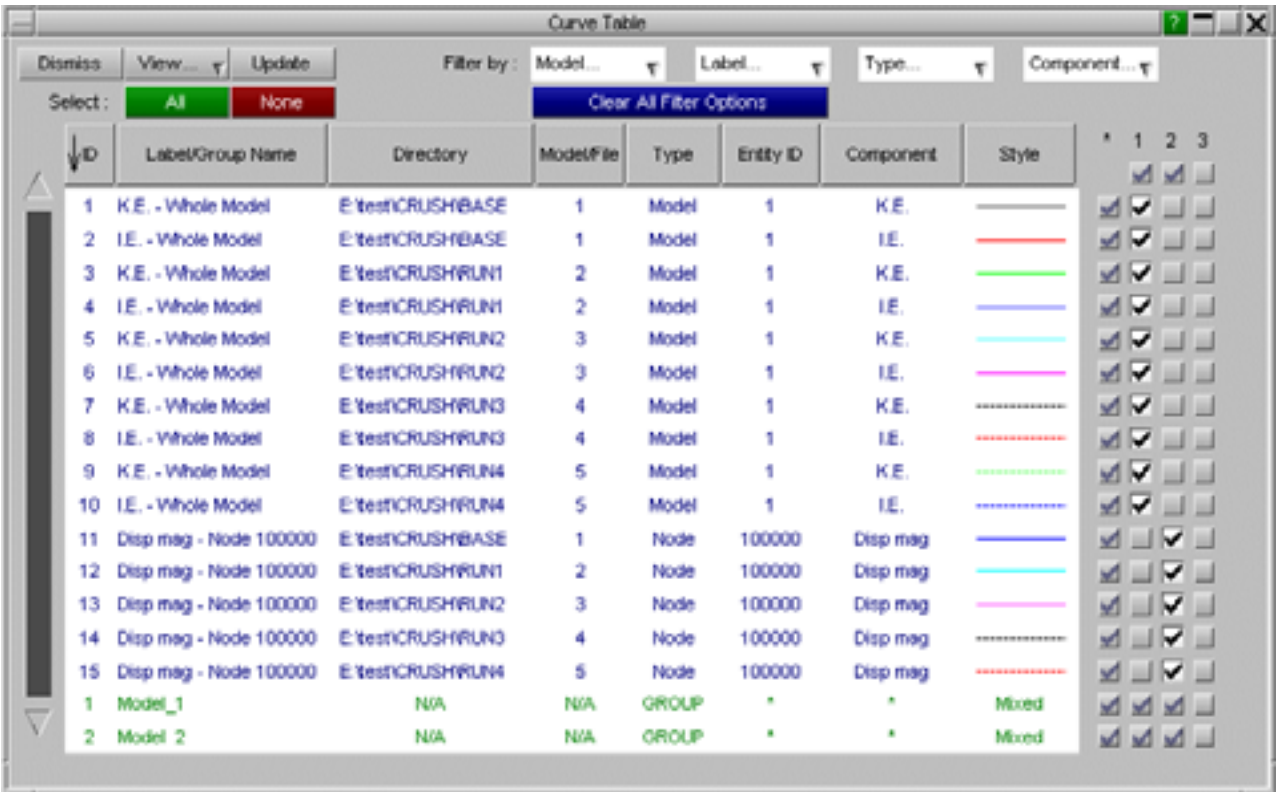
5.3.3 Delete...



- Delete All** Select a groups of curves for deletion
- Curves...** Deletes all current curves. You are prompted for confirmation first!
- Delete by Model...** Delete curves belonging to a Model
- Delete by Group...** Delete curves by Curve Group
- All Blanked** Delete all the curves that are currently blanked
- All Visible** Delete all the curves that are currently unblanked

5.3.4 Table

The Table option can be used to give more control over which curves are blanked and unblanked in all of the currently defined graphs. By default the Curve Table displays a scrolling list of all of the currently defined curves and curve groups along with a set of tick boxes that display the status of the curve in the current graphs. Curves are displayed in BLUE text while curve groups are displayed in GREEN.



For each curve the following information is displayed by default.

ID	Curve ID or Group ID for curve groups
Label	Curve Label or Group Name
Directory	If the curve has been read in from a model then this will be the directory that all the model files are in, if the curve had been read in from a file (.cur. .csv) then this will be the file location. No information is displayed for curve groups.
Model/File	If the curve has been read in from a model then by default this will be the ID of the model. If the curve had been read in from a file then this will be the filename. No information is displayed for curve groups.
Type	The entity type that the curve was generated from. If the curve was read in from a file then this will display "FILE".
Entity ID	ID of the item that the data was read from. If the curve was read from a file then this will be the index within the file for each curve.  If the row represents more than one curve (e.g. curve groups) and the curves have different components then it will display '**'
Component	Data component name.  If the row represents more than one curve (e.g. curve groups) and the curves have different components then it will display '**'
Style	This will show the line colour, style and width used to display the curve.

The column widths can be adjusted by clicking on the bars between the header columns and the order of the columns can be changed by dragging the column headers.

The table contents can also be sorted by any column by clicking on the header button for the column. Clicking on the same header a 2nd time reverses the sort order for the column.

### 5.3.4.1 Adding / Removing Curves from graphs

To add an individual curve (or curve group) to a graph the tick boxes on the right hand side of the curve table can be used.

The first column of tick boxes (under the \*) can be used to add/remove a curve from all the currently defined graphs, while the top row of tick boxes can be used to add/remove all the currently defined curves from a graph.

- If all of the curves are unblanked in a graph then the tick box will display a black tick in a white box.
- If some of the curves are unblanked in a graph then the tick box will display a dark grey tick in a grey box.
- If none of the curves in a group are unblanked in a graph then the tick box will be empty.



Multiple tick boxes can be set/unset by clicking on the first 1 and then using pressing shift which clicking on the last one.

Individual curves can also be selected by clicking on them in the main part of the curve table. Multiple curves can be selected using either CTRL to select a single curve or SHIFT to select a range of curves. As curves are selected they are highlighted in blue and the tick boxes for any unselected curves are greyed out.

When multiple curves have been selected then clicking on a tick box sets the status for all the selected curves.

3	K.E. - Whole Model	E:\test\CRUSH\RUN1	2	Model	1	K.E.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	I.E. - Whole Model	E:\test\CRUSH\RUN1	2	Model	1	I.E.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	K.E. - Whole Model	E:\test\CRUSH\RUN2	3	Model	1	K.E.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	I.E. - Whole Model	E:\test\CRUSH\RUN2	3	Model	1	I.E.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	K.E. - Whole Model	E:\test\CRUSH\RUN3	4	Model	1	K.E.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	I.E. - Whole Model	E:\test\CRUSH\RUN3	4	Model	1	I.E.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

As well as blanking and unblanking curves in graphs a number of other options can be applied to selected curves by right clicking on them.

4	I.E. - Whole Model	E:\test\CRUSH\RUN1	2	Model	4	I.E.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	K.E. - Whole Model	E:\test\CRUSH\RUN2	3	Model				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	I.E. - Whole Model	E:\test\CRUSH\RUN2	3	Model				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	K.E. - Whole Model	E:\test\CRUSH\RUN3	4	Model				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	I.E. - Whole Model	E:\test\CRUSH\RUN3	4	Model				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	K.E. - Whole Model	E:\test\CRUSH\RUN4	5	Model				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	I.E. - Whole Model	E:\test\CRUSH\RUN4	5	Model				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11	Disp mag - Node 100	E:\test\CRUSH\BASE	1	Node				<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
12	Disp mag - Node 100	E:\test\CRUSH\RUN1	2	Node				<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

### 5.3.4.2 View Options

The viewing options popup can be used to control which columns of data are displayed and what items are displayed in the curve table.

By default all 8 columns of information will be displayed, each column can be turned on and off but T/HIS will ensure that at least one column is always displayed.

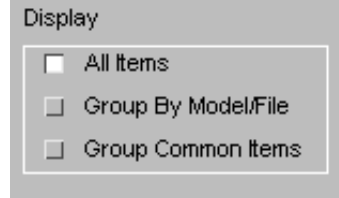
The columns that are initially displayed can be specified in the preference file (see [Appendix H](#) for more details). If the columns are changed then **Save to Pref** can be used to update the preference file.

View...

<b>Select Columns</b> <input checked="" type="checkbox"/> ID <input checked="" type="checkbox"/> Label/Group Name <input checked="" type="checkbox"/> Model/File <input checked="" type="checkbox"/> Type <input checked="" type="checkbox"/> Entity ID <input checked="" type="checkbox"/> Component <input checked="" type="checkbox"/> Style <input checked="" type="checkbox"/> Directory	<b>Display</b> <input type="checkbox"/> All Items <input type="checkbox"/> Group By Model/File <input type="checkbox"/> Group Common Items  <b>Include</b> <input type="checkbox"/> Curves and Group <input type="checkbox"/> Curves Only <input type="checkbox"/> Groups Only	<b>Show Models By</b> <input type="checkbox"/> Model number <input type="checkbox"/> Directory <input type="checkbox"/> THF File <input type="checkbox"/> User Defined
Save to pref	Dismiss	

## Display

This option can be used to control how items are displayed in the curve table.



### All Items

By default the curve table will contain one row for each curve and one row for each curve group.

1	K.E. - Whole Model	E:\BASE	1	Model	1	K.E.	—
2	I.E. - Whole Model	E:\BASE	1	Model	1	I.E.	—
3	K.E. - Whole Model	E:\RUN1	2	Model	1	K.E.	—
4	I.E. - Whole Model	E:\RUN1	2	Model	1	I.E.	—
11	Disp mag - Node 100	E:\BASE	1	Node	10000	Disp ma	—
12	Disp mag - Node 100	E:\RUN1	2	Node	10000	Disp ma	—
1	Model_1	N/A	N/A	GROUP	*	*	Mixed
2	Model_2	N/A	N/A	GROUP	*	*	Mixed

### Group By Model/File

This option will display a single row for all the curves that were read from the same model or file.

*	*	E:\BASE	1	*	*	*	Mixed
*	*	E:\RUN1	2	*	*	*	Mixed
1	Model_1	N/A	N/A	GROUP	*	*	Mixed
2	Model_2	N/A	N/A	GROUP	*	*	Mixed

When this option is selected the columns for curve ID, Label, Type, Entity ID and component display a '\*' as they represent multiple values.

This option can be used to quickly assign all of the curves from a single model or file to the same graph.

### Group By Common Items

This option will display a single row for all the curves that were created using the same entity type, ID and component.

*	*	*	*	Model	1	K.E.	Mixed
*	*	*	*	Model	1	I.E.	Mixed
*	*	*	*	Node	10000	Disp ma	Mixed
1	Model_1	N/A	N/A	GROUP	*	*	Mixed
2	Model_2	N/A	N/A	GROUP	*	*	Mixed

In the example opposite the 1st row represents all of the curves that contain a model Kinetic Energy while the 3rd row represents all the curves that contain a displacement magnitude for Node 10000.

This option can be used to quickly assign all of the curves for the same entity and component to the same graph when comparing results from multiple models.

Include

By default the curve table contains both curves and curve groups. This option can be used to display either just the curves only or just the curve groups.

Include

- ☐ Curves and Group
- ☐ Curves Only
- ☐ Groups Only

Show Models By

If the column displaying the model ID is displayed in the curve table then by default it will display the model number.

This option can be used to display either.

Show Models By

- ☐ Model number
- ☐ Directory
- ☐ THF File
- ☐ User Defined

The model ID	1	K.E. - Whole Model	E:\test\CRUSH\BASE	1	Model	1	K.E.	_____
	2	I.E. - Whole Model	E:\test\CRUSH\BASE	1	Model	1	I.E.	_____
The model directory	1	K.E. - Whole Model	E:\test\CRUSH\BASE	\BASE	Model	1	K.E.	_____
	2	I.E. - Whole Model	E:\test\CRUSH\BASE	\BASE	Model	1	I.E.	_____
The name of the THF file	1	K.E. - Whole Model	E:\test\CRUSH\BASE	base	Model	1	K.E.	_____
	2	I.E. - Whole Model	E:\test\CRUSH\BASE	base	Model	1	I.E.	_____
A user defined model description	1	K.E. - Whole Model	E:\test\CRUSH\BASE	M1	Model	1	K.E.	_____
	2	I.E. - Whole Model	E:\test\CRUSH\BASE	M1	Model	1	I.E.	_____

5.3.4.3 Filter Options

Filter by : 

Model... ▾

Label... ▾

Type... ▾

Component... ▾

Clear All Filter Options

The filter options can be used to filter the list of curves displayed in the curve table.

Multiple filters can be active at the same time

Filter By Model

This option can be used to filter the list of curves by model number. If curves have been read in from a file then an "Other" option will be shown.

In the example opposite only curves that are either from model 1 or model 2 will be displayed.

Model... ▾

Filter by Model

Dismiss

Select : 

All

None

☒ Model 1

☒ Model 2

☐ Model 3

☐ Model 4

☐ Model 5

## Filter By Label

This option can be used to filter the list of curves by label. Up to 5 different strings can be entered and the list of curves displayed will be filtered using those strings. If multiple strings are entered then the strings can either be combined using either "AND" or "OR".

A separate option can be used to ignore the case so that "model" will match both "Model" and "model".

In the example opposite only curves that contain either the word "Model" OR the word "Node" in their labels will be displayed.

Label... ▼

Filter by Label

Dismiss Select : All None ☒ Ignore Case

☒ Model  
☒ Node  
☐  
☐  
☐

☐ And  
☒ Or

## Filter By Type

This option can be used to filter the list of curves by entity type. The list of entity types displayed will automatically update to show the entity types for all the curves that are currently stored in T/HIS.

In the example opposite only curves that contain "Model" data are displayed.

Type... ▼

Filter by Type

Dismiss Select : All None

☒ Model  
☐ Node

## Filter By Component

This option can be used to filter the list of curves by component type. The list of components displayed will automatically update to show the components for all the curves that are currently stored in T/HIS.

In the example opposite only curves that are either Model Kinetic Energy or Nodal Displacement Magnitudes are displayed.

Component... ▼

Filter by Component

Dismiss Select : All None

☒ Model - K.E.  
☐ Model - I.E.  
☒ Node - Disp mag

## Include

By default the curve table contains both curves and curve groups. This option can be used to display either just the curves only or just the curve groups.

Include

☐ Curves and Group  
☐ Curves Only  
☐ Groups Only

## 5.3.5 Summary

Displays a window from which a group of curves may be chosen. The maximum and minimum values of the selected curves are then displayed.

## 5.3.6 List

Displays a **LIST CURVES** window, from which a number of curves may be selected. The data point values for the selected curves are then listed in a listing box.



### 5.3.7 Scan

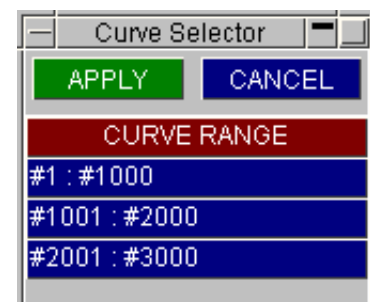
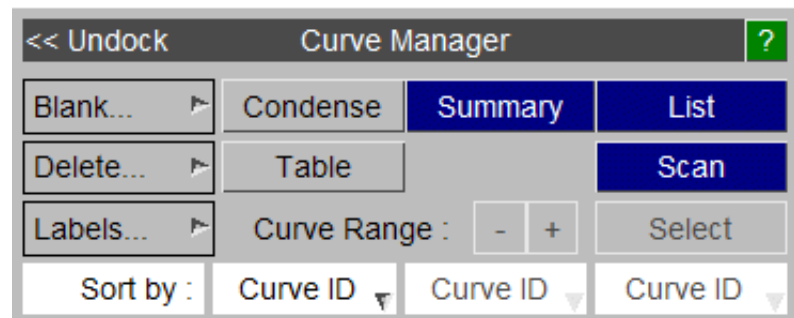
Displays a window from which a group of curves may be chosen. The maximum and minimum values of the selected curves are then displayed.

### 5.3.8 CURVE RANGE SELECTION

The range buttons in the Curve Control menu can be used to when you are working with more than 1000 curves to move between groups of 1000 curves. Pressing the green + tab will display the next group of 1000 curves in the menu, whilst pressing the red - tab will display the previous group of 1000 curves.

Alternatively pressing the Select button will bring up the following new window.

Select the appropriate group of 1000 curves and press apply to display those 1000 curves in the Curve control menu.



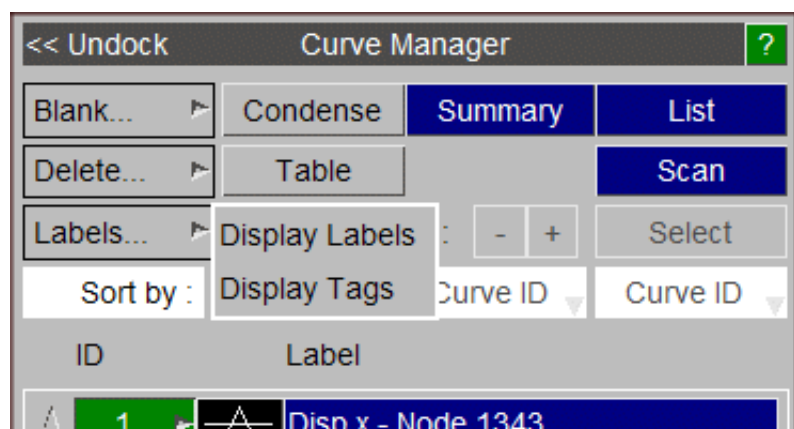
### 5.3.10 CURVE TAGS

Curves can be given tags to act as internal identifiers within T/HIS which can be used to reference curves in order to perform operations on them.

In order to display the curve tags, toggle on the Show Labels arrow and select Show Tags. The tag names can be defined in the input boxes.

When a curve file is written, T/HIS will save the tags of all the tagged curves in the file.

When performing operations in the dialogue box, curves can be referenced by their tags. The tag must be placed in double quotes.

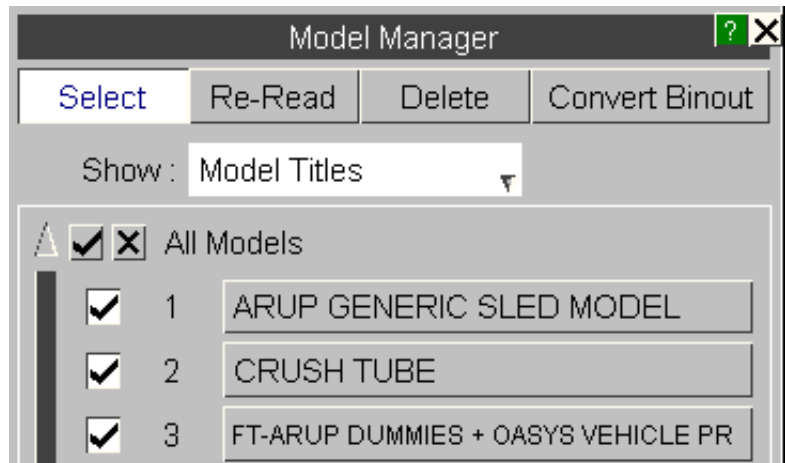


```
Operate > dif "Vel_x_n_123" #2050
(Written [Analysis Velocity (Dif)] to curve #2050)
(DIF #1002 => #2050)
```

## 5.4 Model Manager

### 5.4.1 Select

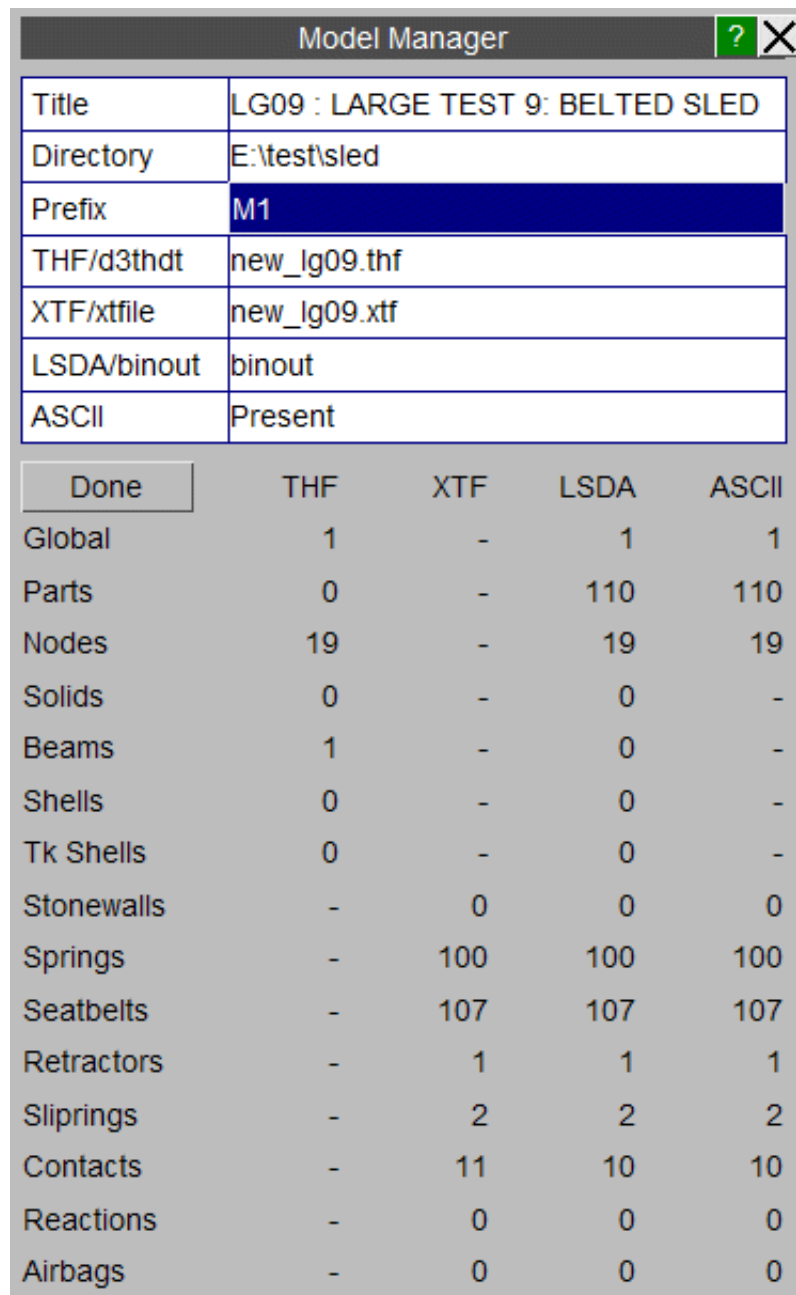
This allows the user to turn models on/off. Deselecting a model will result in removal of its entities as options when reading data. Models can be displayed according to their titles or alternatively by the directories they were read in from.



Clicking on the button displaying a model title will produce a menu similar to that illustrated. The number of each type of item in the model and the sources T/HIS found for that item type's data will be shown. The user can select which file type is preferred for the data for each type of item (see [Preferences](#)).

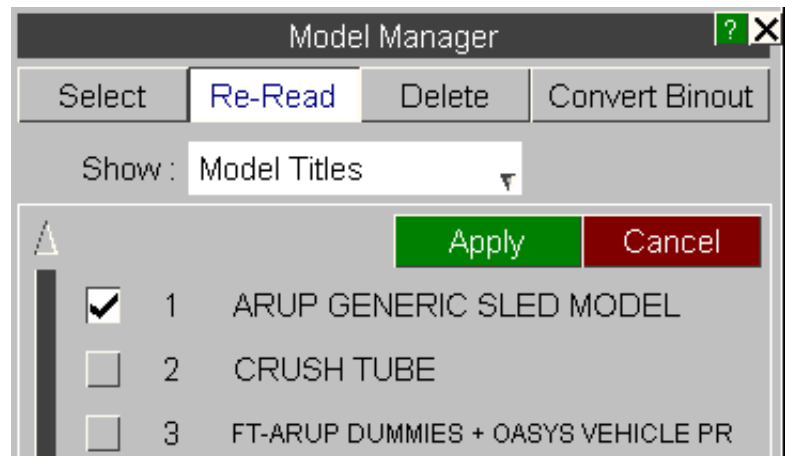
#### Prefix

This menu can also be used to define a user defined model prefix. This prefix can be added automatically to the start of curve labels to help identify which model they belong to.



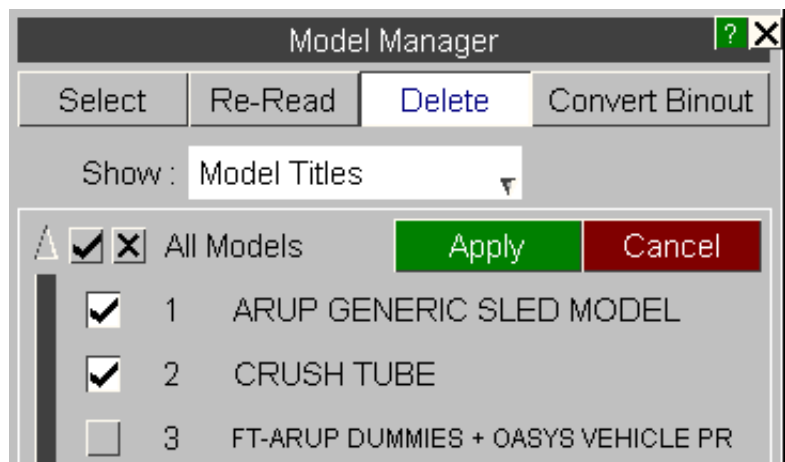
### 5.4.2 Re-Read

The re-read option can be used to rescan and update the model. This will find any new data written to disk since the file was last read.



### 5.4.3 Delete

This option allows the user to select and delete models from T/HIS. Any curves that have been read in from a model that is deleted are NOT deleted with the model. Any number of models to be deleted from T/HIS.

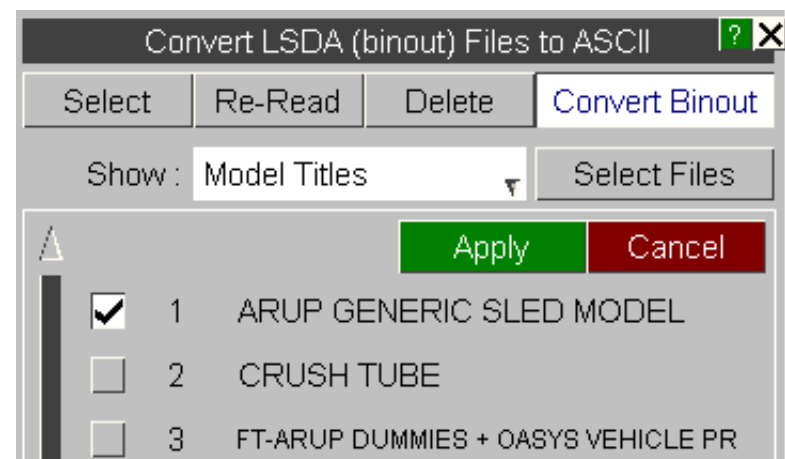


### 5.4.4 Convert Binout

This option can be used to convert LSDA binout files into the older ASCII files. The menu allows a number of models to be selected.

The **Select Files** button allows the user to specify which ASCII files are to be created.

All of the ASCII files are written into the directory containing the LSDA file.



## 5.5 EDIT Options

This menu allows you to examine and make modifications to the curve data points. You are always working on a "scratch" copy of the curve. The permanent curve is only updated when you SAVE it explicitly.

Moving around the curve data is done through the use of scroll bars on the data panel.

### Save

Saves the edited curve as either a new curve or overwrites the original.

### Restart

Resets the curve being edited to the values at the start of the edit session.

### Quit

Quits the curve editor without making any changes to the curve

### Labels...

Allows the title, axis and line label to be changed (see [Section 5.5.3](#) for more details)

### Replace

Allows curve values to be changed by overtyping the x and y values.

### Insert Before

Inserts a new point in the curve before the selected point.

### Insert After

Inserts a new point in the curve after the selected point.

### Delete

Deletes the selected point.

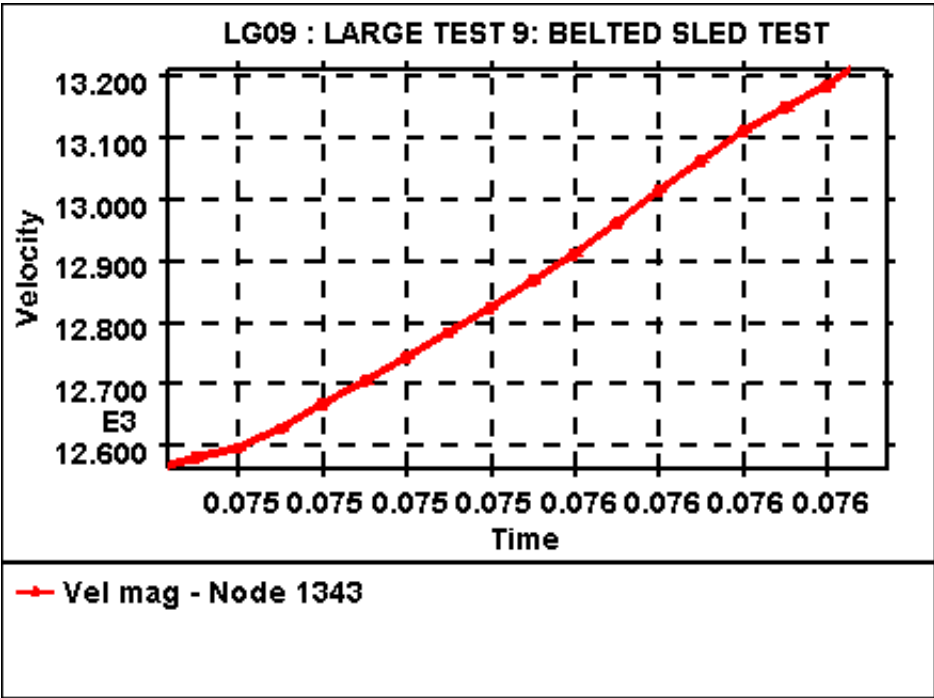
The screenshot shows the 'Curve Editor' window. At the top, there are buttons: 'Save' (highlighted in red), 'Restart', 'Quit', and 'Labels...'. Below these are 'Replace', 'Insert Before', 'Insert After', and 'Delete'. A status bar shows '#points: 1001' and 'Command :'. Below the status bar is an 'Undo' button. The main area is a table with 22 rows, each representing a point. The columns are 'Point', 'X value', and 'Y value'. The 'X value' column contains values in scientific notation, and the 'Y value' column contains values in scientific notation. A vertical scroll bar is on the left of the table.

Point	X value	Y value
1	0.000000E+00	0.000000E+00
2	9.940239E-05	1.035303E+00
3	1.977450E-04	2.124691E+00
4	2.990677E-04	3.401202E+00
5	3.974104E-04	4.798676E+00
6	4.987331E-04	6.375494E+00
7	5.970757E-04	7.994342E+00
8	6.983984E-04	9.703846E+00
9	7.997211E-04	1.141409E+01
10	8.980638E-04	1.304722E+01
11	9.993864E-04	1.468456E+01
12	1.097729E-03	1.621935E+01
13	1.199052E-03	1.773808E+01
14	1.297394E-03	1.914959E+01
15	1.398717E-03	2.054395E+01
16	1.497060E-03	2.185106E+01
17	1.598382E-03	2.316937E+01
18	1.699705E-03	2.448088E+01
19	1.798048E-03	2.576451E+01
20	1.899370E-03	2.710555E+01
21	1.997713E-03	2.841944E+01
22	2.099036E-03	2.976754E+01

The **Command** text-box allows control by command line (see [Section 5.5.2](#) for more details).

### 5.5.1 Interactive Curve Editing

After a curve has been selected it is displayed using a thicker line to highlight it in any graphs that it is visible in.



As well as being highlighted the curve points can be edited interactively and the Quick Pick menu in the main Tool Bar (see [Section 6.1](#) for more details) is replaced with the EDIT menu.

Modify (Drag X,Y) ▾

Edit Points

Modify (Drag X,Y) ... ▾

Insert (Between) ... ▾

Delete (Point) ... ▾

Locate

Restart

Save (Overwrite)

Save (Highest used + 1)

Quit

#### 5.5.1.1 Modify

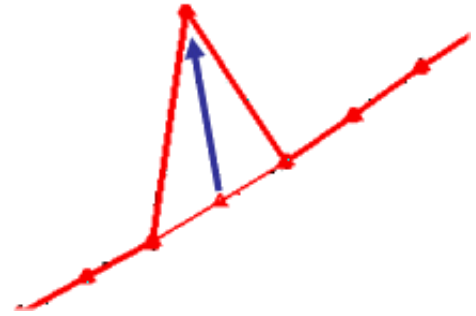
Modify (Drag X,Y) ... ▾

Drag X,Y

Drag X

Drag Y

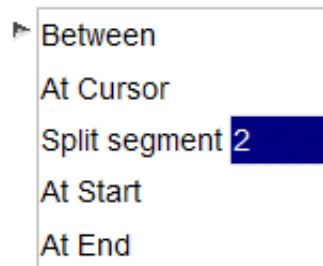
**Drag X,Y** Modify the point nearest to the screen pick by dragging it's position in both the X and Y axis directions.



**Drag X** Drag a point in the X axis direction only.

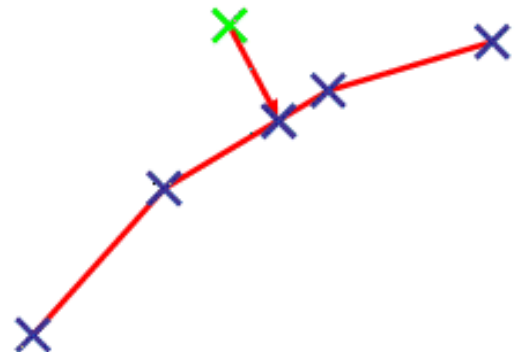
**Drag Y** Drag a point in the Y axis direction only.

#### 5.5.1.2 Insert Insert (Between) ...



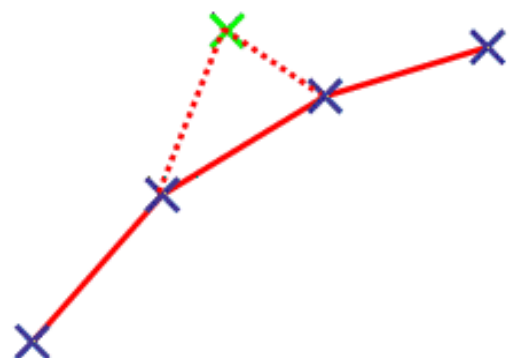
##### **Between**

Finds the nearest segment to the point selected on the screen and then projects the point onto the segment.



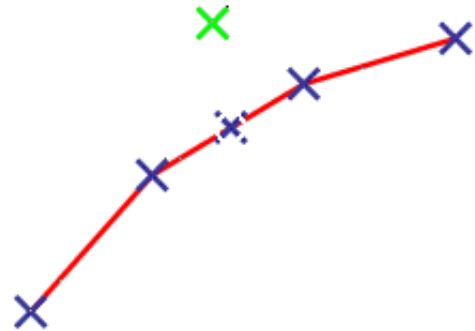
##### **At Cursor**

Finds the nearest segment to the point selected on the screen and then inserts the a point at the screen location between the 2 ends of the segment.



**Split Segment**

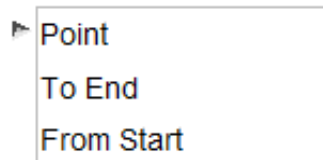
Finds the nearest segment to the point selected on the screen and then splits the segment in 2 or more parts.

**At Start**

Inserts a new point at the screen location before the first point in the curve.

**At End**

Inserts a new point at the screen location after the last point in the curve.

**5.5.1.3 Delete** Delete (Point) ...**Point**

Finds the nearest point to the screen pick and deletes it.

**To End**

Finds the nearest point to the screen pick and deletes all points in the curve from that point onwards.

**From Start**

Finds the nearest point to the screen pick and deletes all points in the curve up to that point.

**5.5.1.4 Locate**

Finds the nearest point to the screen pick and updates the list of points in the main edit panel so that the points either side of the picked point are displayed.

**5.5.1.5 Restart**

Resets the curve being edited to the values at the start of the edit session.

**5.5.1.6 Save (Overwrite)**

Overwrite the original curve with the edited one.

**5.5.1.7 Save (Highest used + 1)**

Save the edited curve as a new curve without overwriting the original curve.

### 5.5.1.8 Quit

Quits the curve editor without making any changes to the curve.

## 5.5.2 Command line mode

In command line mode editing of curves is done in a similar fashion using the following commands.

<b>Moving around the curve:</b>	<b>F</b>	<b>Forward</b>	Move forward 16 lines
	<b>B</b>	<b>Back</b>	Move back 16 lines
	<b>T</b>	<b>Top</b>	Move to the top of the curve
	<b>E</b>	<b>End</b>	Move to the end of the curve
	<b>N</b>	<b>Number</b>	Move to given line number
<b>Modifying the curve:</b>	<b>Cn</b>	<b>Change</b>	Change line n
	<b>In</b>	<b>Insert</b>	Insert points before line n
	<b>An</b>	<b>Append</b>	Append points after line n
	<b>D n1 n2</b>	<b>Delete</b>	Delete lines n1 to n2
	<b>L</b>	<b>Label</b>	Change the line label
	<b>R</b>	<b>Reset</b>	Reset the curve back to the original curve
<b>Saving and Plotting the curve:</b>	<b>W</b>	<b>Write</b>	Write the curve
	<b>S</b>	<b>Save</b>	As write
	<b>PE</b>	<b>Plot Edited</b>	Plot the edited curve
	<b>PA</b>	<b>Plot All</b>	Plot the edited and original curve
	<b>PL</b>	<b>Plot</b>	Plot the current T/HIS curves
	<b>Q</b>	<b>Quit</b>	Quit the editor

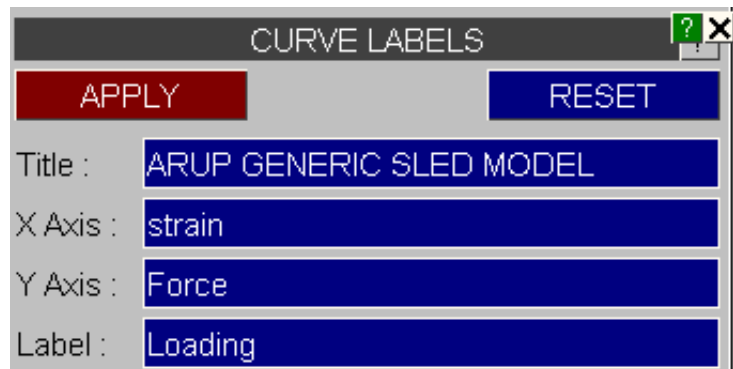
In command line mode the EDIT menu is reached by typing **/ED**



## 5.5.3 Curve Labels

Each curve has four labels associated with it:

Title	The title string at the top of the plot
X label	The label for the X axis of the plot
Y label	The label for the Y axis of the plot
Label	The label applied to the line itself



The screenshot shows a dialog box titled "CURVE LABELS". At the top right of the dialog is a green square with a white question mark and a close button (X). Below the title bar are two buttons: "APPLY" (red) and "RESET" (blue). Underneath these buttons are four text input fields, each preceded by a label: "Title : ARUP GENERIC SLED MODEL", "X Axis : strain", "Y Axis : Force", and "Label : Loading".

The first three are only used on a plot if this curve is the first (or only) curve to be plotted, and the relevant labels are in "automatic" mode (see [TITLE and AXIS](#)).

You can change any of these by simply overtyping whatever is currently there. When you are happy with the result use the **APPLY** button to dismiss this box, saving the new values. The labels here are scratch values, current only in this editor, the permanent curve labels are only overwritten with them if you **SAVE** this edited curve.

**RESET** will restore the scratch labels to the original values of the permanent curve being edited.

The title, axis and line labels can also be modified using the [dialogue box](#)

## 5.6 LINE STYLES

The **LINE STYLE** menu is shown in the figure (right). This menu can be used to change the colour, width, style and symbol for any of the curves that are currently being used.

When a curve file is written, T/HIS will save the line style for each curve in the file.

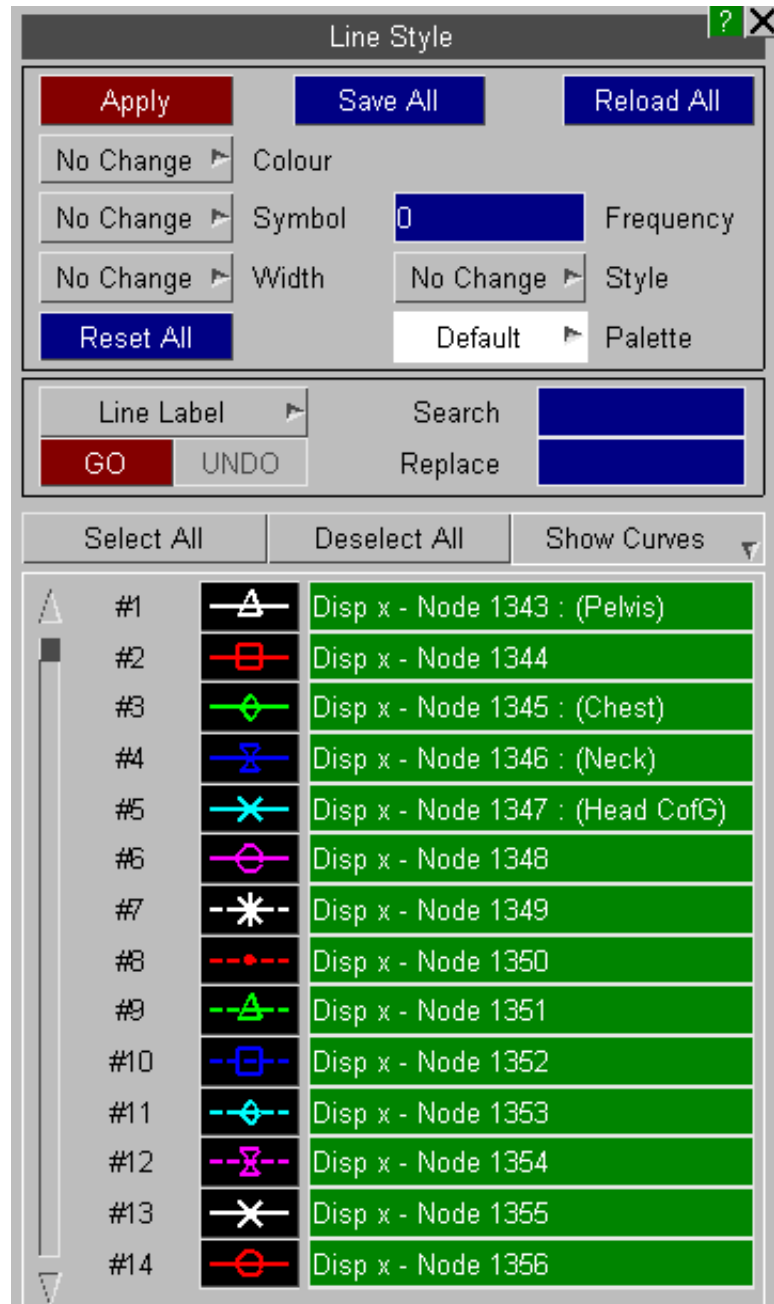
The lower half of this panel contains a list of all the curves that are currently being used. By default the curve that was clicked on in the **CURVE CONTROL** menu will be highlighted and the colour and symbol buttons in the top section of the menu will show the setting for that curve.

The **SAVE...** button can be used to save the current set of line styles to a file while the **RELOAD...** button can be used to reload a set from a previously saved file. The **DEFAULT** button will reset all the curve styles to the original T/HIS settings.

If you wish to modify the colour/style of more than one curve at a time additional curves may be selected by pressing the button next to each curve number that depicts the current line style.

**SELECT\_ALL** and **DESELECT\_ALL** may be used to select/deselect all the curves.

Line Styles can also be edited using the [dialogue box](#)



### 5.6.1 APPLY

This button will **APPLY** the current line colour, symbol, width and style selection to all the curves that have been selected.

## 5.6.2 COLOUR

Pressing the right mouse button while over the colour button will invoke a colour popup menu.

T/HIS has a built in palette of 30 predefined colours and 6 user defined colours. Colours are defined using 6 digit Hexadecimal values using the format RRGGBB.

RR Red Component (0-255)  
GG Green Component (0-255)  
BB Blue Component (0-255)



Colour ID	Name	Alternative Name	Value
1	COL 1	WHITE	FFFFFF
2	COL 2	RED	FF0000
3	COL 3	GREEN	00FF00
4	COL 4	BLUE	0000FF
5	COL 5	CYAN	00FFFF
6	COL 6	MAGENTA	FF00FF
7	COL 7	YELLOW	FFFF00
8	COL 8	ORANGE	FF9C00
9	COL 9	TURQUOISE	21FF94
10	COL 10	INDIGO	7B00FF
11	COL 11	LIME	BDF339
12	COL 12	SKY	39BDF3
13	COL 13	PINK	FF7B7B
14	COL 14	BLACK	000000
15	COL 15	PALE YELLOW	FFFF9C
16	COL 16	GOLD	FFCE00
17	COL 17	OLIVE	7B7B00
18	COL 18	DARK MAGENTA	9C3163
19	COL 19	MEDIUM GREEN	9CCE00
20	COL 20	MEDIUM BLUE	7B7BFF
21	COL 21	HOT PINK	FF9CCE
22	COL 22	LIGHT PINK	FFCE9C
23	COL 23	SEA GREEN	317B63
24	COL 24	MAROON	7B0000
25	COL 25	DARK GREEN	007B00
26	COL 26	PURPLE	7B007B
27	COL 27	NAVY	00007B
28	COL 28	DARK GREY	393939
29	COL 29	MEDIUM GREY	7B7B7B
30	COL 30	LIGHT GREY	BDBDBD
31	COL 31	USER 1	-
32	COL 32	USER 2	-
33	COL 33	USER 3	-
34	COL 34	USER 4	-
35	COL 35	USER 5	-
36	COL 36	USER 6	-

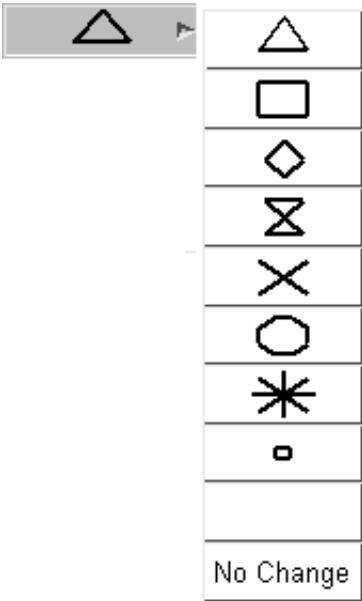
As well as the 36 colour options **Foreground** and **Background** can be selected to change the colour to the **Foreground** and **Background** colours defined in the [Display](#) menu.

If **N/C** is selected then the **Apply** button will have no effect on the colour of the currently selected curves..

### 5.6.3 SYMBOL

Pressing the right mouse button while over the **Symbol** button will invoke a symbol popup menu that allow any of the 9 T/HIS symbols to be selected (the 9<sup>th</sup> is a blank symbol that can be selected so that a curve can be plotted without a symbol). As well as the 9 symbols the menu also contains a "No Change" option.

The **Symbols Frequency** controls how often a symbol is drawn on a curve. By default, symbols are not drawn; they can be switched on using the [Display](#) menu.



### 5.6.4 WIDTH

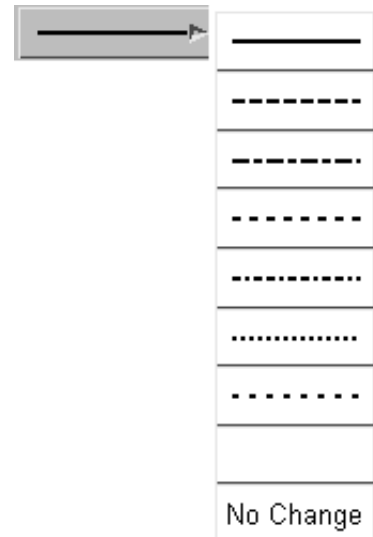
Pressing the right mouse button while over the width button will invoke a popup menu that allows 4 different line widths to be selected or "No Change".



## 5.6.5 STYLE

Pressing the right mouse button while over the style button will invoke a popup menu that allows 8 different line styles to be selected (the 3<sup>rd</sup> is actually a blank line that can be selected so that a curve can be plotted without a line).

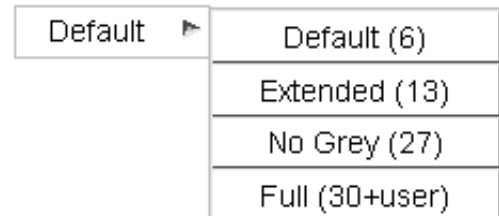
As well as the 8 line styles the menu also contains a "No Change" option.



## 5.6.6 CURVE PALETTE

By default T/HIS uses 6 colours (White, Red, Green, Blue, Cyan and Magenta) for any curves that have not had a colour explicitly defined for them. Curves 1,7,13... will be White, 2,8,14... will be Red.

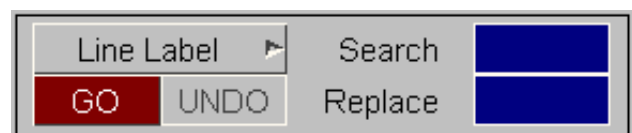
This option can be used to change the default number of colours T/HIS uses.



Default	Use the default 6 colours
Extended	Use the first 13 colours
No Grey	Use all 30 predefined colours except the 3 grey ones
Full	Use all 30 predefined colours plus any user defined ones.

The default value for the curve palette can also be specified in the "preferences" file (see [Appendix H](#) for more details).

## 5.6.7 MODIFYING LABELS



Multiple curve labels may be edited using the Search and Replace option to enter the string to search for and the string to replace it with. ^ can be used to insert text at the beginning of a label while \$ can be used to append to the end of a label. The table below shows the effect of 2 search and replace examples.

	Example 1	Example 2
Original Label	Displacement N1034	Time
Search String	N1	\$
Replace String	Node 1	(s)
Modified Label	Displacement Node 1034	Time(s)

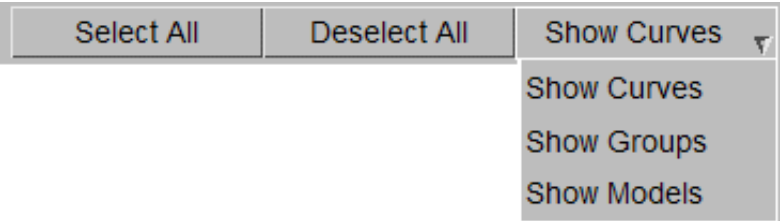
The **GO** button will initiate the search and replace on all the curves that are currently selected (highlighted in the bottom half of the menu), while the **UNDO** button can be used to reset the labels to what they were before the search and replace.

Pressing the right mouse button while over the **Line Label** button will invoke a popup menu that allows the label that is being modified to be swapped between the **Line Label**, **X-Axis Label** and the **Y-Axis Label**.

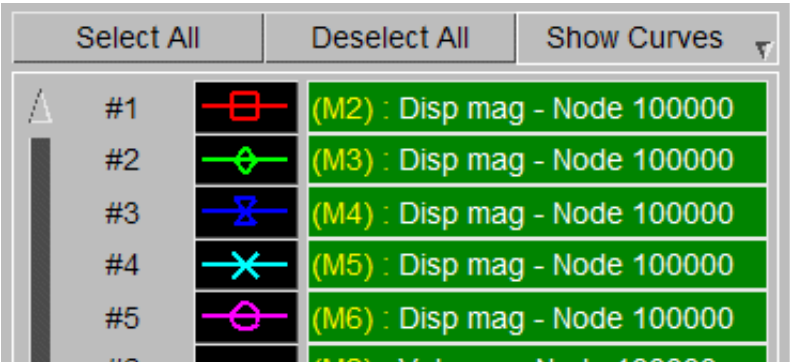
Line labels can also be modified by using the [dialogue box](#)



5.6.8 **SELECTING CURVES**

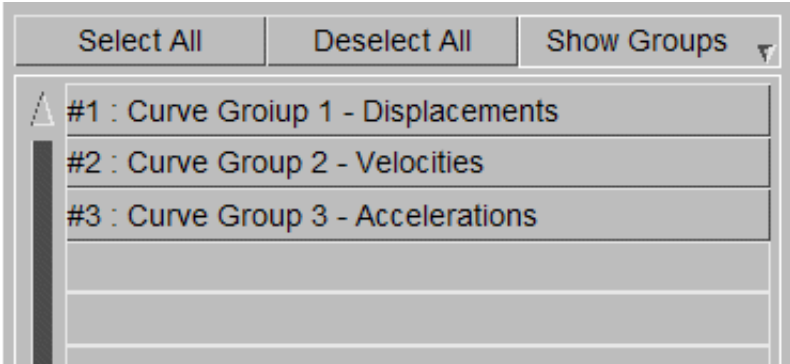


By default the Style menu will display a list of all the current defined curves so that the style for individual curves can be



Instead of displaying individual curves the style menu can be changed to display a list of any currently defined curve groups.

If curve groups are selected then the style will be applied to all of the curves in the curve group.



The style menu can also display a list of all the models currently loaded in T/HIS.

If models are selected then the style will be applied to any curve that was created using data from the model.



## 5.6.9 LINE STYLE EDITING IN THE DIALOGUE BOX

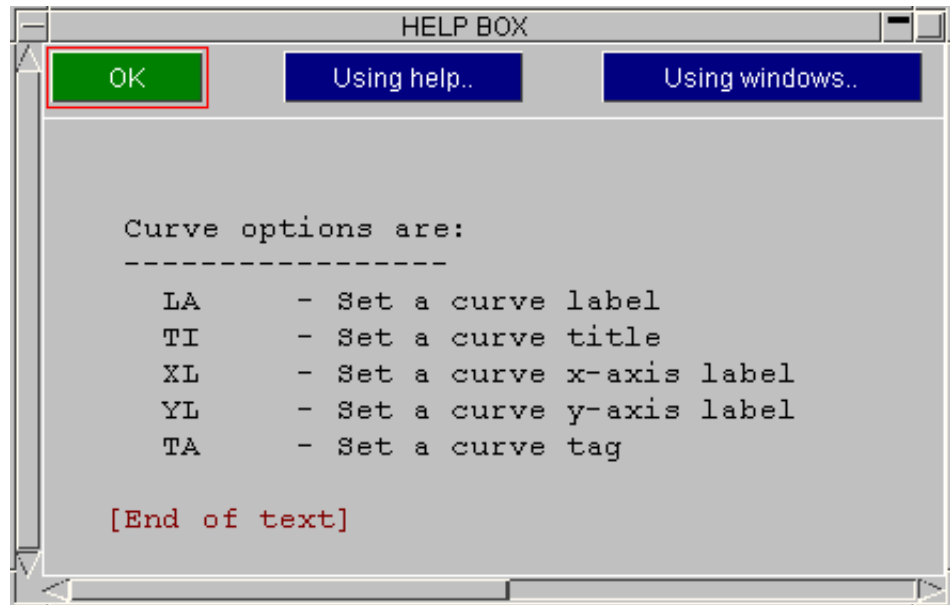
The dialogue box can be used to edit curve styles

To access this feature, enter the command /style at the Command prompt

Enter M at the STYLE > command prompt for a list of all available dialogue box commands

The following commands are available:

SET  
READ  
WRITE  
DEFAULT  
FIX  
GM



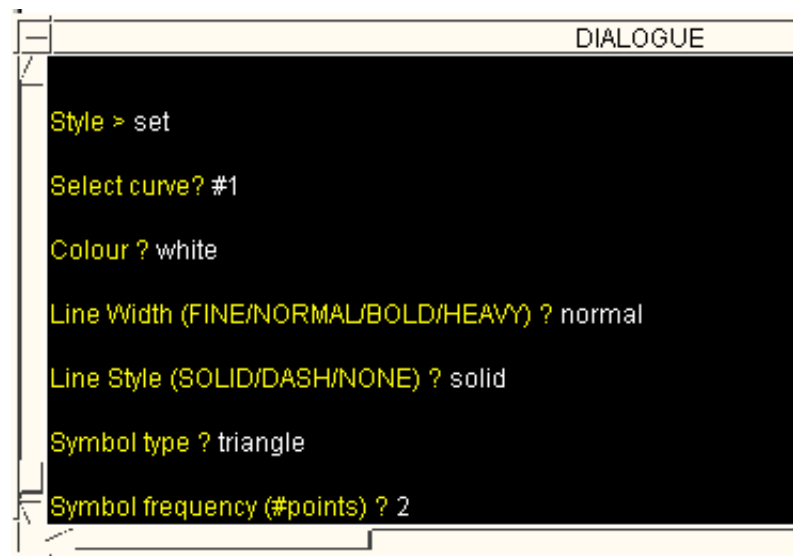
### SET

This option allows the user to set the style properties for individual curves.

Enter the curve number (e.g #1 for curve 1) at the Select Curve? command prompt.

T/HIS will prompt the user to input the desired style properties in the order:

Colour; Enter the colour for the line  
Line Width; Enter the desired line width for the line  
Line Style; Enter the desired line style (e.g. dashed) for the line  
Symbol Type; Enter the desired Symbol Type  
Symbol Frequency; Enter the desired frequency of the symbols in the format



### READ

This option allows the user to read a style file containing style information and apply that style to a particular curve

Enter the name of the style file at the Style File? command prompt.

### WRITE

This option allows the user to write a style file containing style information.

### DEFAULT

This option allows the user to reset all the curve styles to the default settings.

**FIX**

This is an **ON/OFF** switch which resets the curve styles when they are plotted on the screen so that the curves cycle through the default T/HIS colours and styles as they are plotted. This will result in the first curve being plotted always being white, the second red, the third green, etc regardless of their curve numbers. The default is **OFF**.

**GM**

This option will display the Global Menu in a separate window

## 5.6.10 LABEL AND TITLE EDITING IN THE DIALOGUE BOX

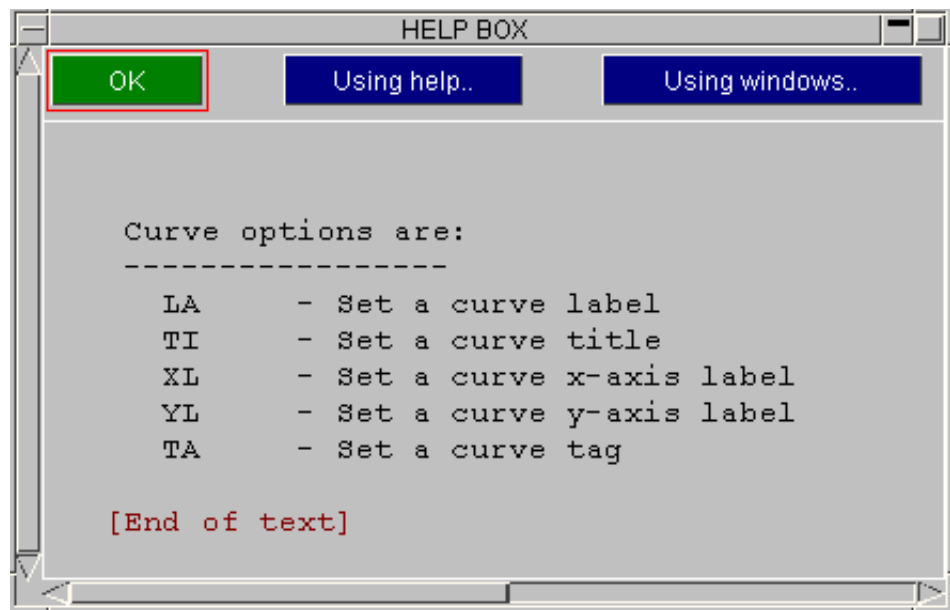
The dialogue box can be used to edit curve labels, x-axis and y-axis labels and curve titles

To access this feature, enter the command /cur at the Command prompt

Enter M at the CURVE > command prompt for a list of all available dialogue box commands

The following commands are available:

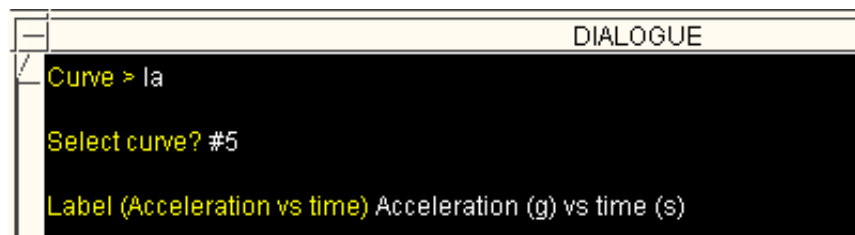
LA  
TI  
XL  
YL  
TA

**LA**

This option allows the user to edit the label for individual curves.

Enter the curve number at the Select curve? prompt

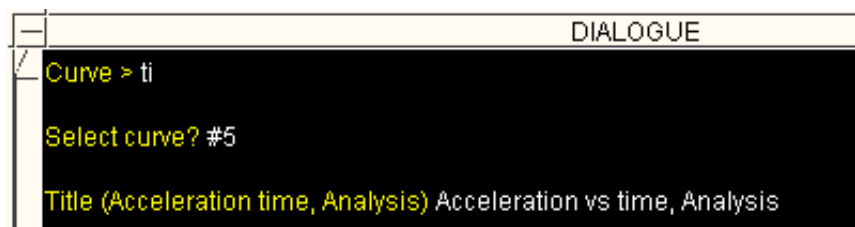
Enter the desired new label at the Label prompt, The current Label will be displayed in brackets

**TI**

This option allows the user to edit the title for individual curves.

Enter the curve number at the Select curve? prompt

Enter the desired new title at the Title prompt, The current title will be displayed in brackets





**XL**

This option allows the user to edit the x-axis label for individual curves.

Enter the curve number at the Select curve? prompt

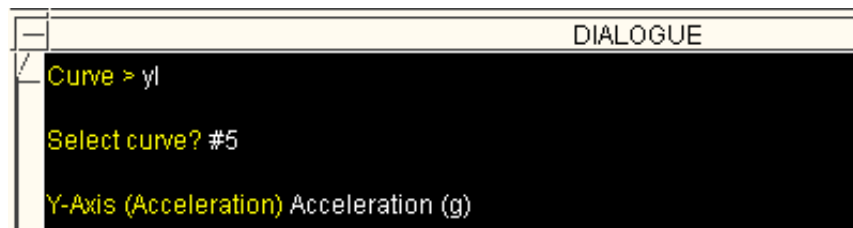
Enter the desired new title at the X-Axis prompt, The current x-axis label will be displayed in brackets

**YL**

This option allows the user to edit the y-axis label for individual curves.

Enter the curve number at the Select curve? prompt

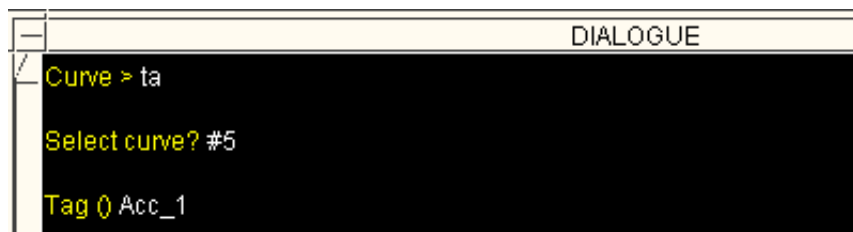
Enter the desired new title at the Y-Axis prompt, The current y-axis label will be displayed in brackets

**TA**

This option allows the user to edit the tag for individual curves.

Enter the curve number at the Select curve? prompt

Enter the desired new Tag at the Tag prompt, The current tag will be displayed in brackets



## 5.7 Command / Session Files

Command and session files are used to drive or record a T/HIS session. Both session (save) and command (playback) files have been set up to act like tape recorders; and the concept of "recording" and "playing back" files will be used below.

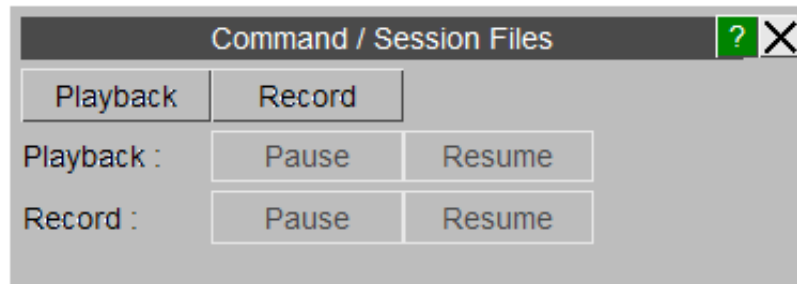
These files ("button click" command files) are not easy to edit by hand and they are not always backwards compatible between versions of T/HIS. For these reasons most users prefer the new [FAST-TCF](#) format, which can also be recorded and [played back](#) from within T/HIS.

In screen menu mode a command has a meaning beyond the simple command word. For example, **HELP** appears in many different places, with a distinct meaning (or relevance) in each place. Therefore, context information is stored when saving screen menu session files.

In practice the following information is saved:

- the command itself - whether typed or inferred from a button
- the button identification (if any)
- the parent window identification
- the menu item (if relevant)
- the action type (screen pick, button press, etc)
- any x/y coordinates that may be relevant.

A choice of either writing ("recording") session files or executing ("playing back") command files is given. By default commands are not saved. If they are to be saved the session file record switch must be turned on.



## 5.7.1 Writing ("Recording") Session Files

To write a session file the record **CONTROLS...** button must be pressed, displaying the **RECORD COMMAND FILES** menu shown right.

Pressing the **RECORD >** button will start the session file. Thereafter, all commands (except those in the session/playback windows) are saved in an internal scratch file. In order to save these commands to disk they must be written explicitly using the **SAVE TO DISK** button. They can then be read back in and replayed

A variety of features are available to help move around the file. These are shown in the **FILE POSITION AND CONTROL** area of the panel. The file can be indexed at particular user defined points using the **INDEX MARKS** menu is accessed by pressing the **INDEX...** button. These may be used as targets of a search and also to control recording.

The scratch file is random access, and can be moved back and forth and reviewed at will. To help with this it is possible to switch between **RECORD** and **REVIEW** modes in the session file control box:

**RECORD** records all your commands when running

**REVIEW** plays back your recorded commands

A command file can be stepped through or run backwards or forwards. It may also be searched for a particular command. As with a real tape recorder, if the pointer is moved backwards and recording continued the commands that were previously stored will be overwritten from that point.

The session file recording and command file playback operations are totally separate: they can be thought of as two separate tape recorders. As a consequence it is possible to record commands that are being played back: in effect it is possible to edit and combine files.

## 5.7.2 Executing ("Playing Back") Command Files

As above, the **PLAYBACK COMMAND FILES** menu, shown right, must be invoked from the **COMMAND/SESSION FILES** window.

This is done by pressing the playback **CONTROLS...** button. An existing file must then be read. This is analogous to loading a tape into the tape recorder: it is then converted into an internal scratch format (random access, as above) and can be played back or previewed at will.

Once a file is read in either **PLAYBACK** or **PREVIEW** mode may be selected:

**PLAYBACK** actually executes the commands, **PREVIEW** simply lists them without executing them.

The file may be stepped through backwards or forwards at will, and searches made for commands. Playback commences at the current line when **PLAY** is pressed, so it is possible to skip unwanted commands or repeat a sequence.

As with **RECORD** above, index marks can be inserted, which may be used as targets of a search and also to stop playback.

Command / Session Files

Playback

Record

Playback :

Pause

Resume

Record :

Pause

Resume

Filename and mode

File:

E:\roger.tcf

☐ Preview mode

☐ Playback mode

REREAD FILE

DELETE FILE

File position and control

<< SEARCH

PLAY >

SEARCH >>

< STEP

STOP

STEP >

TOP

Goto line:

END

INDEX...

SPEED...

ERRORS...

Current command status

Line no:

2 / 3

Box name:

Top menu box

Function:

Button Press

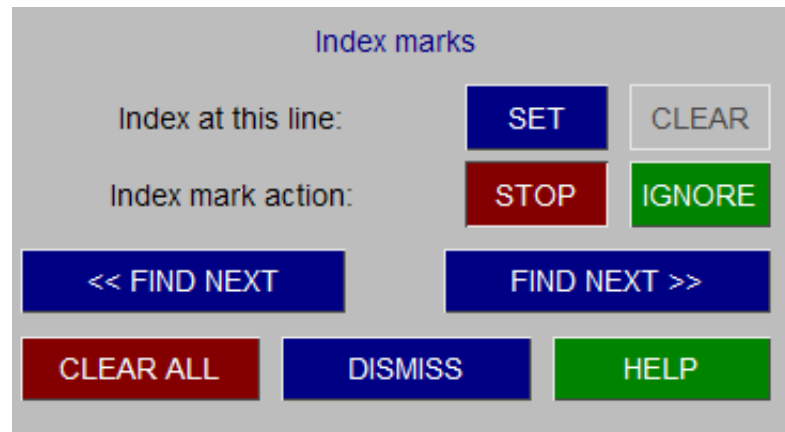
Command:

Command File

### 5.7.3 INDEX MARKS

"Index marks" are optional flags that you can set at any line in a file. They are not interpreted as commands but rather treated as markers which are used as targets of [SEARCH](#) operation. Index mark functions are:

<b>SET</b>	Set an index mark on this line;
<b>CLEAR</b>	Clear an index mark set on this line
<b>STOP</b>	Stop in PLAY/REVIEW mode when index found
<b>IGNORE</b>	Ignore index marks during PLAY/REVIEW
<b>FIND INDEX</b>	Finds the next index mark: "<<" searching backwards, ">>" searching forwards
<b>CLEAR ALL</b>	Clear all index marks in the file

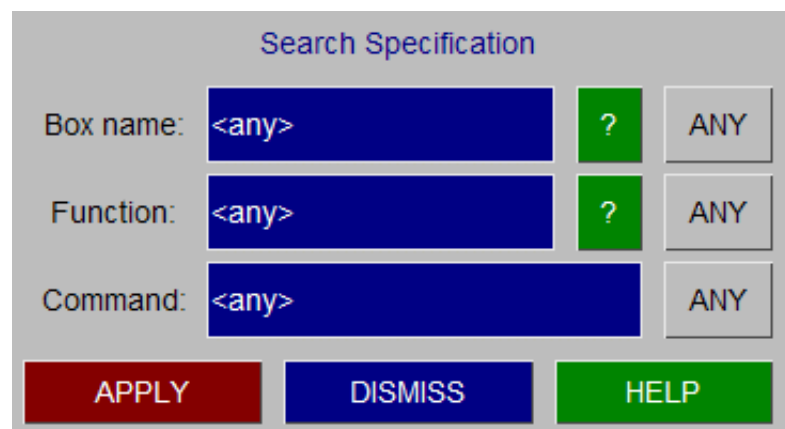


### 5.7.4 SEARCH

The **SEARCH** buttons can be used to find a specific command when in **REVIEW** mode. You can search through the command file for a match to any permutation of the following:

<b>Box name</b>	The name of a screen menu box inside' which an event occurred
<b>Function</b>	The screen menu function type. This is "button press", "dialogue", etc;
<b>Command</b>	The command word(s) to look for.

The default for all of these is "<any>", ie a wildcard search, but you can specify a value by typing into the appropriate text box. When you have filled in all the fields you need, press **APPLY** to start the search. "Box name" and "Function" fields are unlikely to be of use to most users, you can list all valid events using "?" button to provide a menu to pick from. The **ANY** button may be used for any field to restore it to its default (wildcard) status.



### 5.7.6 Command Line Mode Session / Command File Control

The available features in command line mode for command and session file control are very basic. A session file can be recorded at any point by typing **SF** (in the **GLOBAL MENU**) followed by the desired filename. This is equivalent to the **RECORD** button in screen menu mode. The session file can be closed by typing **CS** and is automatically written to disk. This is equivalent to pressing the **STOP** and **SAVE TO DISK** buttons in screen menu mode.

To execute an existing command file in T/HIS simply type **CF**, followed by the filename.

No previewing/reviewing or editing of command/session files is possible in command line mode.

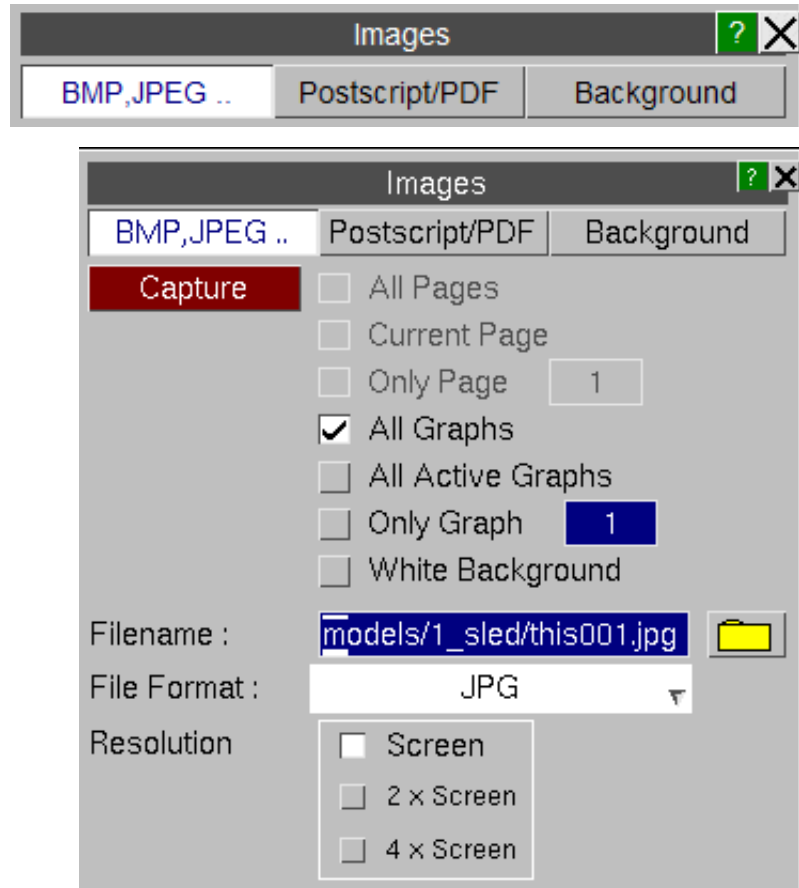
### 5.7.7 Command Files From Earlier Versions Of T/HIS

Command files recorded in Version 9.0 or earlier will not work in T/HIS 9.4.

## 5.8 IMAGE Options

### 5.8.1 BMP,JPEG ...

This menu can be used to save an image containing one or more graphs in a number of different formats.



#### All Pages

Each page will be saved as a single image to multiple files. The filenames used will be based on the filename selected by the user..

*This option will only be available if T/HIS contains multiple graphs on more than one page ([see section 3.2](#)).*

#### Current Page

A single image containing currently displayed page will be generated.

*This option will only be available if T/HIS contains multiple graphs on more than one page ([see section 3.2](#)).*

#### Only Page (n)

A single image containing the selected page will be generated.

*This option will only be available if T/HIS contains multiple graphs on more than one page ([see section 3.2](#)).*

#### All Graphs

A single image will be generated containing all of the graphs.

*This option will only be available if T/HIS only contains a single page ([see Section 3.2](#)).*

#### All Active Graphs

A single image will be generated containing all of the currently active graphs.

*This option will only be available if T/HIS only contains a single page ([see Section 3.2](#)).*

#### Only Graph (n)

A single image containing the selected graph will be generated.

#### White Background

Captures the image with a white background and black foreground. Once the image is captured the colours are reset to their original values.

### 5.8.1.1 File Format

#### 8-bit file formats

<b>BMP Uncompressed</b>	Uncompressed 8 bit Microsoft windows bitmap. The approximate size of the file (in bytes) is  file size= image width * image height
<b>BMP Compressed</b>	8 bit RLE Microsoft windows bitmap.
<b>PNG</b>	8 bit Portable Network Graphics
<b>GIF</b>	Graphics Interchange Format

#### 24-bit file formats

<b>BMP</b>	Uncompressed 24 bit Microsoft windows bitmap. The approximate size of the file (in bytes) is  file size = 3 * image width *image height
<b>PNG</b>	24 bit Portable Network Graphics
<b>JPG</b>	JPEG (Joint Photographic Experts Group) file
<b>PPM</b>	Uncompressed Portable PixMap. The approximate size of the file (in bytes) is  file size = 3 * image width *image height

#### 8 bit BMP (Uncompressed) ▾

8-bit file formats
BMP (Uncompressed)
BMP (Compressed)
PNG
GIF
24-bit file formats
BMP
PNG
JPG
PPM

### 5.8.1.2 Resolution

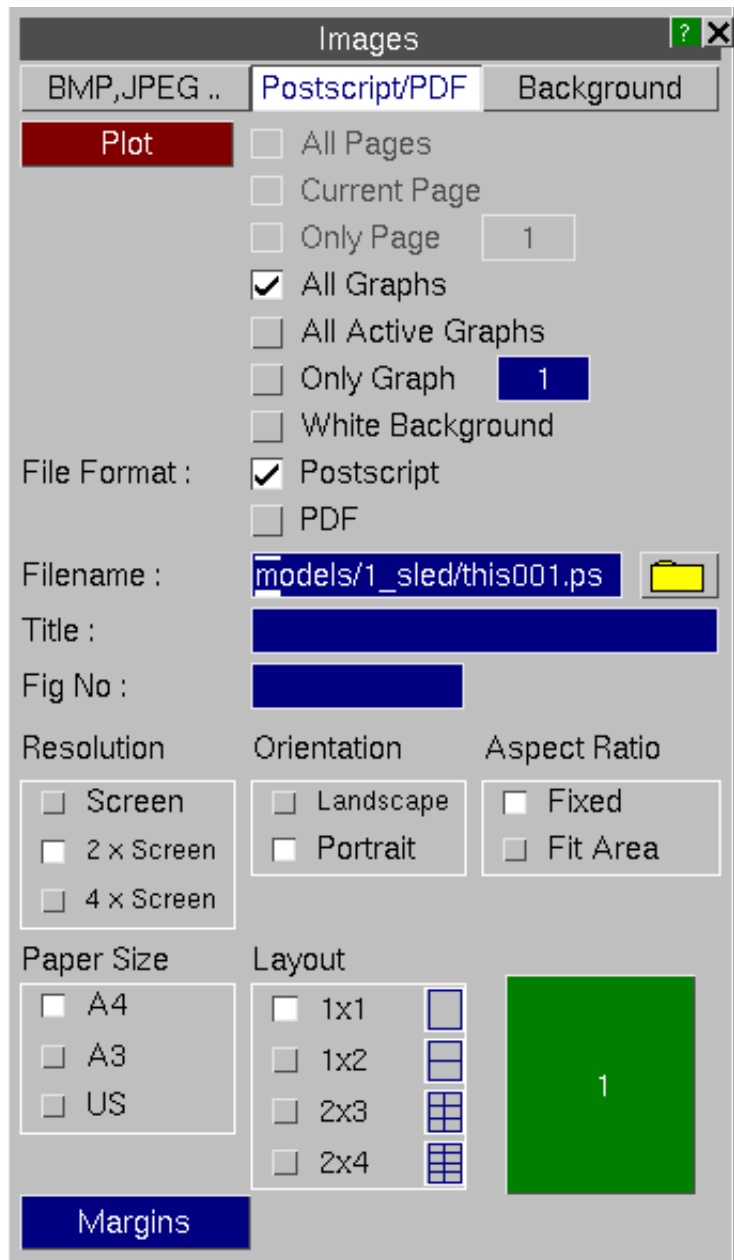
All images can be output at either the screen resolution or at a resolution of either 2 or 4 times the screen resolution.

<input type="checkbox"/>	Screen
<input type="checkbox"/>	2 x Screen
<input type="checkbox"/>	4 x Screen

## 5.8.2 Postscript

This menu can be used to save an image containing one or more graphs to either a PDF or Postscript file.

All PDF and Postscript files are generated using raster images so that the contents of the screen is exactly reproduced.



### All Pages

All T/HIS pages containing 1 or more graphs will be saved to a single file.  
*This option will only be available if T/HIS contains multiple graphs on more than one page (see Section 3.2).*

### Current Page

The current T/HIS page will be saved.

*This option will only be available if T/HIS contains multiple graphs on more than one page (see Section 3.2).*

### Only Page (n)

A single image containing the selected page will be generated.

*This option will only be available if T/HIS contains multiple graphs on more than one page (see Section 3.2).*

### All Graphs

A single image will be generated containing all of the graphs.

*This option will only be available if T/HIS only contains a single page (see Section 3.2).*



**All Active Graphs**

A single image will be generated containing all of the currently active graphs.

*This option will only be available if T/HIS only contains a single page ([see Section 3.2](#)).*

**Only Graph (n)**

A single image containing the selected graph will be generated.

**White Background**

Captures the image with a white background and black foreground. Once the image is captured the colours are reset to their original values.

**5.8.2.1 File Format**

All images can be output at either the screen resolution or at a resolution of either 2 or 4 times the screen resolution.

☒ Postscript  
☐ PDF
**5.8.2.2 Title and Fig Number**

By default PDF and Postscript files are not labeled and have no figure number, but you may add either or both of these. They are always put at the bottom of each page, along the short edge, regardless of the orientation used for plots.

Test output 1

1.1

**5.8.2.3 Resolution**

All images can be output at either the screen resolution or at a resolution of either 2 or 4 times the screen resolution.

☐ Screen  
☐ 2 x Screen  
☐ 4 x Screen
**5.8.2.4 Orientation**

All images can be output in either landscape or portrait format.

☐ Landscape  
☐ Portrait
**5.8.2.5 Aspect Ratio**

By default all images are output using a fixed aspect ratio. This option can be used to stretch each image to fit the available space on the page. Different scaling factors will be applied to the horizontal and vertical directions and the image will be distorted.

☐ Fixed  
☐ Fit Area
**5.8.2.6 Paper Size**

The paper size can be set to be either A4 (210 x 296mm), A3 (296 x 420mm) or US (letter - 216 x 279mm). The default size is A4.

☐ A4  
☐ A3  
☐ US
**5.8.2.7 Layout**

Multiple plots on a page are also available. In landscape format there is a choice of 1, 2x2, 3x3 and 4x4 plots to a page. In portrait format there is a choice of 1, 1x2, 2x3 and 2x4 plots on a page. By default there is a single plot on a page.

When multiple plots are requested the order in which they are performed can be defined.

☐ 1x1
 ☐

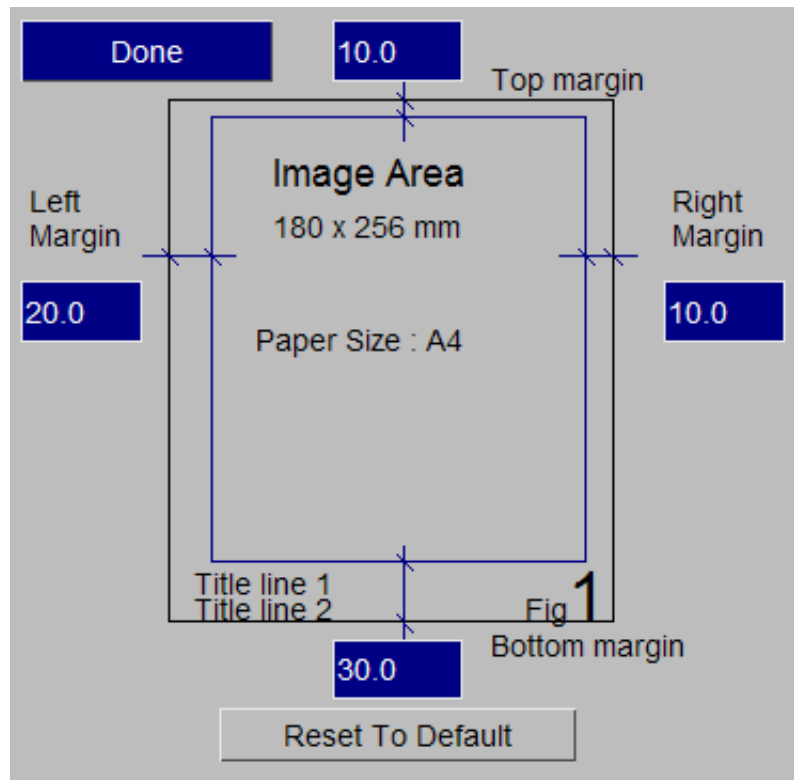
☐ 1x2
 ☐

☐ 2x3
 ☐

☐ 2x4
 ☐

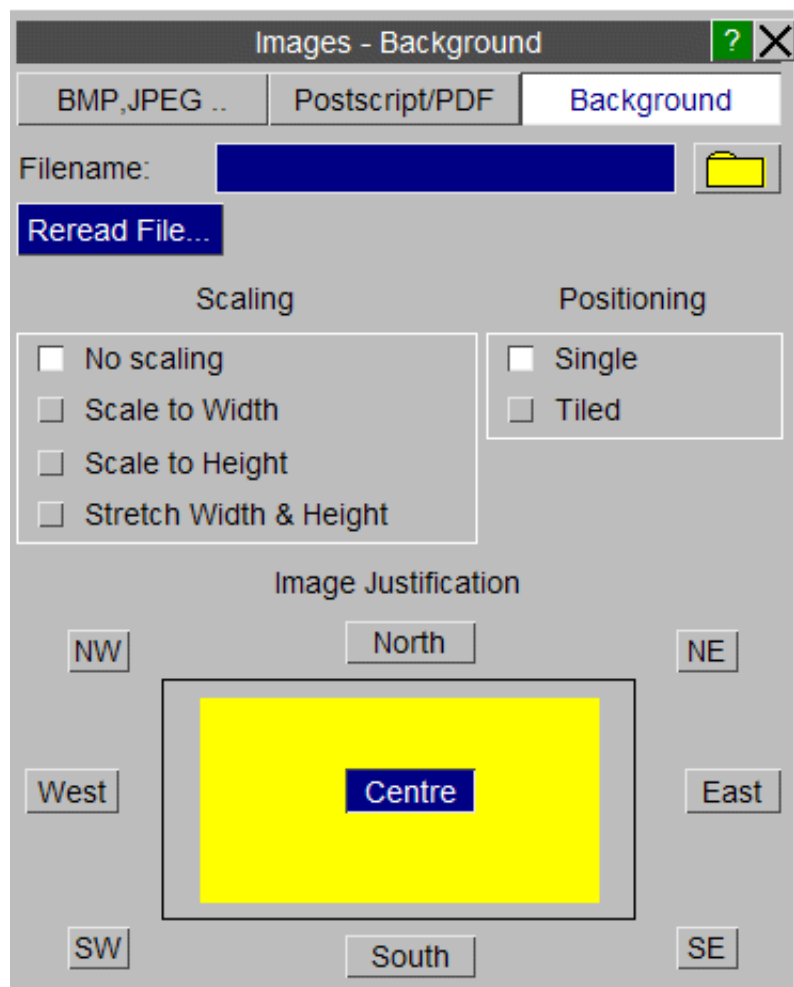
### 5.8.2.8 Margins

The Margins can be used to change the top, bottom, left and right margins for each page.



### 5.8.3 Background

This option can be used to add a background image to each graph (see section [5.16.8](#) for more details).



## 5.9 OPERATE Options

The **OPERATE** menu shown in the figure (right). If the mouse is left hovering over an option a short description of the function will appear. For these functions, the user selects a range of curves to be operated on. A range may be one or more curves, making it possible to operate on multiple curves, for example add 20 curves to 20 curves.

ABS	ADD (y)	ADD (x)	AVE	CAT	CLIP
COM	DIF	DIV (y)	DIV (x)	ENV	ERR
INT	LSQ	MAP	MAX	MIN	MON
MUL (y)	MUL (x)	NOR (y)	NOR (x)	ORDER	REC
RES	REV	R-AVE	SMO	SQR	STRESS
SUB (y)	SUB (x)	SUM	TRA	VEC	VEC(2D)
WINDOW	ZERO	dB	dBA	Octave	

The options with the **OPERATE** menu are split into 3 groups. The first group require 2 sets of curves as input. The second group require a single set of curves as input. The third group also require a single set of curves as input but the output from these functions is a single curve. (See [Section 5.0](#) for more information on curve groups).

**5.9.1 ABS** Produces the absolute y-values of a curve.

**5.9.2 ADD** Add the y axis values together for two curves or add a constant value to all the y-values. If two curves are being added together they must have identical x-axis values.

**5.9.3 ADX** Add the x axis values together for two curves or add a constant value to all the x-values. If two curves are being added together they must have identical y-axis values.

**5.9.4 AVE** Produces a single curve that is the average of the input curves.

**5.9.5 CAT** Concatenate the second curve to the end of the first.

### 5.9.6 CLIP

Clip a curve to remove any points that exceed a set of specified minimum and maximum x & y axis value. The user is prompted for minimum and maximum values after the curves have been selected..

Instead of typing in values for the limits individual x and y axis minimum and maximum values can be selected by picking screen points. In addition to picking individual points an area can be dragged out interactively to set all 4 limits.

When picking screen points the default is to allow any point to be selected.

**Snap to curve points** can be used to select the point on the nearest curve instead of the screen coordinates.

-0.10000E+21	X minimum value	Pick Xmin
0.10000E+21	X maximum value	Pick Xmax
-0.10000E+21	Y minimum value	Pick Ymin
0.10000E+21	Y maximum value	Pick Ymax
<input type="checkbox"/> Snap to curve points		Select by area

### 5.9.7 COM

Two curves are combined to give a new curve. For example if a displacement/time curve is combined with a velocity/time curve a velocity/displacement curve will result. If the 2 curves do not contain points at the same x values then the curve with the larger x-axis intervals is automatically mapped on to the x-axis values of the other curve.

If the curves do not start and finish at the same x-axis values then only the points for which the two curve x-axes overlap are mapped onto each other.

### 5.9.8 DIF

A curve is differentiated with respect to the x-axis variable.

### 5.9.9 DIV

Divide the y axis values of the first curve by the y axis values of the second curve (or a constant). If two curves are being used they must have identical x-axis values.

### 5.9.10 DIX

Divide the x axis values of the first curve by the x axis values of the second curve (or a constant). If two curves are being used they must have identical y-axis values.

### 5.9.11 ENV

Produces a single curve that bounds the maximum and minimum values of the group of input curves.

**5.9.12  
ERR**

This option reports the degree of correlation between 2 input curves. The first curve selected is used as a reference curve and the following parameters are then reported :

Maximum difference :	Value & Time Value as a %age of reference curve Value as a %age of reference curve peak value.
Average difference -	Value %age of reference curve peak value
Area Weighted Difference Correlation Parameter -	0 to 1

For more details on this function please see [Appendix G](#)

**5.9.13 INT** A curve is numerically integrated with respect to the x-axis variable using Simpson's rule.**5.9.14  
LSQ** Fits a straight line through the points using the least squares method.**5.9.15  
MAP** The second curve is mapped onto the first curve, the resulting curve has identical x-axis values to the reference (first) curve with y-axis values obtained from the mapped (second) curve.**5.9.16  
MAX** Produces a single curve that bounds the maximum values of the group of input curves.**5.9.17 MIN** Produces a single curve that bounds the minimum values of the group of input curves.**5.9.18  
MON** Sorts a curve into monotonically increasing x-axis values.**5.9.19  
MUL** Multiply the y axis values together for two curves or multiply all the y-values by a constant. If two curves are being multiplied together they must have identical x-axis values.**5.9.20  
MUX** Multiply the x axis values together for two curves or multiply all the x-values by a constant. If two curves are being multiplied together they must have identical y-axis values.**5.9.21  
NOR** Normalize a curve so that the y axis values lie in the range [-1, +1].**5.9.22  
NOX** Normalize a curve so that the x axis values lie in the range [-1, +1].**5.9.23  
ORDER** Reverse the order of all the points in the curve.**5.9.24  
REC** Produces the reciprocal of the y-values of a curve.**5.9.25  
RES** Calculate the vector magnitude from a group of input curves.

### 5.9.26 REV

Reverses the x and y axes of a curve. For example if you start with a curve with displacement (y axis) against time (x axis) you end up with a curve of time (y axis) against displacement (x axis).

### 5.9.27 R-AVE

Produces a single curve of the running average on the input curve.

0.0000

Averaging Window

If the time window is set to 0 then the y values for the output curve are the average value of all the point up to that point.

If the time window is non-zero (T) then the y values at each point are calculated by averaging the values between  $-T/2$  and  $+T/2$ .

### 5.9.28 SMO

A moving average technique is used to smooth (filter) a curve. The user will be prompted for a smoothing factor.

Smoothing Factor &gt; 1 (integer)

7

The integer refers to the number of points included in the averaging of each point. The value you want will depend on the number of points in the curve and the amount of smoothing required. A certain amount of trial and error is necessary to get the required result.

### 5.9.29 SQR

Take the square root of the y-values of a curve.

### 5.9.30 STRESS

Converts a stress / strain curve between True and Engineering Stress /Strain.

### 5.9.31 SUB

Subtract the y axis value (or constant) of the second curve from the first curve. If two curves are being subtracted they must have identical x-axis values.

### 5.9.32 SUX

Subtract the x axis value (or constant) of the second curve from the first curve. If two curves are being subtracted they must have identical y-axis values.

### 5.9.33 SUM

Calculates the sum of a group of curves. This "sums" up the y-axis values of a group of curves, and maps the result onto the x-axis of the first curve.

### 5.9.34 TRA

Translate a curve with respect to the x and y axes. The user is prompted for the x and y values.

X Translation 0.0000

Y Translation 0.0000

### 5.9.35 VEC

Calculate the vector magnitude from three input curves.

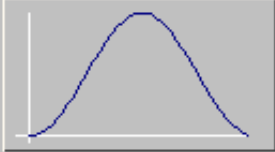
### 5.9.36 VEC(2-D)

Calculate the vector magnitude from two input curves.

### 5.9.37 WINDOW

This function is typically used to modify a curve before carrying out an FFT on it.

The y-axis values for each of the input curves is multiplied by a factor between 0 and 1. Three different window shapes are available. The **Store Window Curve** option can be used to output the multiplying factors to a separate curve if required.

<input type="checkbox"/> Hanning		
<input type="checkbox"/> Cosine Taper		
<input type="checkbox"/> Exponential		
<input type="checkbox"/> Store Window Curve		
10.000	%age Lead-in	

### 5.9.38 ZERO

Translate a curve so that the first data point is moved to (0,0).

By default this option will translate the curve in both X and Y, alternatively the curve can be translated in X only or Y only.

<input type="checkbox"/> Zero X and Y
<input type="checkbox"/> Zero X only
<input type="checkbox"/> Zero Y only

### 5.9.39 dB

Converts a curve to dB.

$$f(x) = 20\log(y/\text{ref})$$

1.0000	Reference Value
--------	-----------------

### 5.9.40 dBA

Converts a curve from dB to dBA by applying "A" weighting factors to the curve values.

- Narrow band A weighting values are calculated using a formula.
- 1/3 Octave A weighting values are calculated from a lookup table.

<input type="checkbox"/> Use "narrow band" A weighting
<input type="checkbox"/> Use 1/3rd Octave A weighting

### 5.9.41 Octave

Converts a curve from narrow band to either Octave bands or 1/3 rd Octave bands.

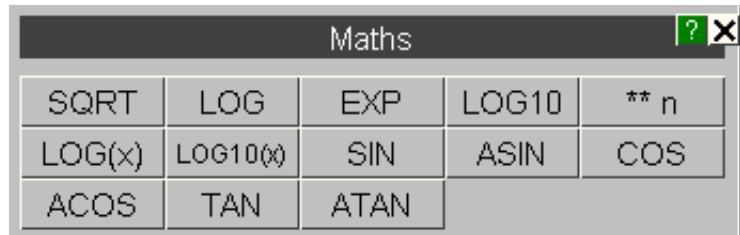
The input curve can either be a curve that has already been converted to dB or it can be an unconverted "linear" curve.

The output curve can also be generated using either Mean values or RMS values.

<input type="checkbox"/> 1/3rd Octave Bands	<input type="checkbox"/> Generate RMS values
<input type="checkbox"/> Octave Bands	<input type="checkbox"/> Generate Mean values
<input type="checkbox"/> Linear Input	
<input type="checkbox"/> dB Input	

## 5.10 MATHS Options

The **MATHS** menu is shown in the figure (right). This menu allows mathematical operations to be applied to curves. These options are self explanatory and work on the y-value of the curve (except where explicitly stated).



**Note:** Trigonometrical functions expect the user to work in radians.

- |                        |                                |
|------------------------|--------------------------------|
| <b>5.10.1 SQRT</b>     | The square root of a curve.    |
| <b>5.10.2 LOG</b>      | Natural log (to base e)        |
| <b>5.10.3 EXP</b>      | e to power of.                 |
| <b>5.10.4 LOG10</b>    | Log to base 10                 |
| <b>5.10.5 **n</b>      | Raise to power n.              |
| <b>5.10.6 LOG(x)</b>   | Log to base 10 (x-axis values) |
| <b>5.10.7 LOG10(x)</b> | Log to base 10 (x-axis values) |
| <b>5.10.8 SIN</b>      | Sine (radians assumed)         |
| <b>5.10.9 ASIN</b>     | Arc sine                       |
| <b>5.10.10 COS</b>     | Cosine                         |
| <b>5.10.11 ACOS</b>    | Arc cosine                     |
| <b>5.10.12 TAN</b>     | Tangent                        |
| <b>5.10.13 ATAN</b>    | Arc tangent                    |



# 5.11 AUTOMOTIVE Options

The **AUTOMOTIVE** menu is shown in the figure (right). The automotive options are a number of operations that can be performed on curves, typically finding their use in the Automotive industry. They consist of filters and injury criteria calculations, along with a number of other useful functions.

<< Undock Automotive ? X					
C60	C180	C600	C1000	BUT	BUT(p)
FIR	HIC	HIC(d)	3ms CLIP	EXC	VC
ASI	THIV	NIJ	TTI	NOR (y)	NOR (x)
REG	VEC	VEC(2d)	ACU	COR1	COR2
COR3	WIF				

All the options in the **AUTOMOTIVE** menu require a single set of curves as input except the **VEC** and **VEC(2D)** options which require groups of 3 or 2 curves respectively as input but only output a single curve. (See [Section 5.0](#) for more information on curve groups).

## Notes on using the various filters

When filtering curves the sampling rate of the data should be considered: it should be at least twenty times the filter cutoff frequency if good results are to be obtained.

T/HIS will reject attempts to filter curves for which the sampling rate is too low, if this happens the **REG** option can be used to increase the number of points. This will allow the filter to function although it is not a good substitute for obtaining data at a higher sampling rate.

For more information on the filters and injury criteria calculations see [Appendices D & G](#).

All of the filters expect the input curve to have a consistent time interval. When using one of the filter options the user can specify a time interval for the curve to be automatically regularised to ( **REG** ) before filtering if the time interval is not consistent. The user can set a default time interval for regularising the input curves in the PREFERENCE menu. The PREFERENCE menu can also be used to automatically convert the x axis values from milliseconds to seconds before filtering and to convert the curve back to milliseconds afterwards.

5.11.1 **C60** Filter a curve using a standard SAE Class 60 filter.

5.11.2 **C180** Filter a curve using a standard SAE Class 180 filter.

5.11.3 **C600** Filter a curve using a standard SAE Class 600 filter.

5.11.4 **C1000** Filter a curve using a standard SAE Class 1000 filter.

5.11.5 **BUT** The curve is passed through a Butterworth filter. The user is prompted for the cutoff frequency and the order of the filter.

Cut-off frequency (Hz)	1000.0
Filter order (integer)	1

5.11.6 **BUT(p)** This passes a curve through a Pure Butterworth filter. This is the same as the BUT function above, but the two refinements, described in [Appendix D](#), to minimise end-effects and phase change errors are not included.

5.11.7 **FIR** Special filter for US "SID" dummy.

## 5.11.8 HIC

Calculates the Head Impact Criteria from an acceleration time history. The user is prompted for the time window and the acceleration conversion factor.

Normally this option writes the HIC value to the screen. If required the values may also be written out to a file using the **WRITE TO FILE** option.

The time unit for the input curve should be seconds. T/HIS look at the range of the x-axis values and if the range is >1 then T/HIS will assume the x-axis values are in ms and it will automatically divide the x-axis values by 1000.

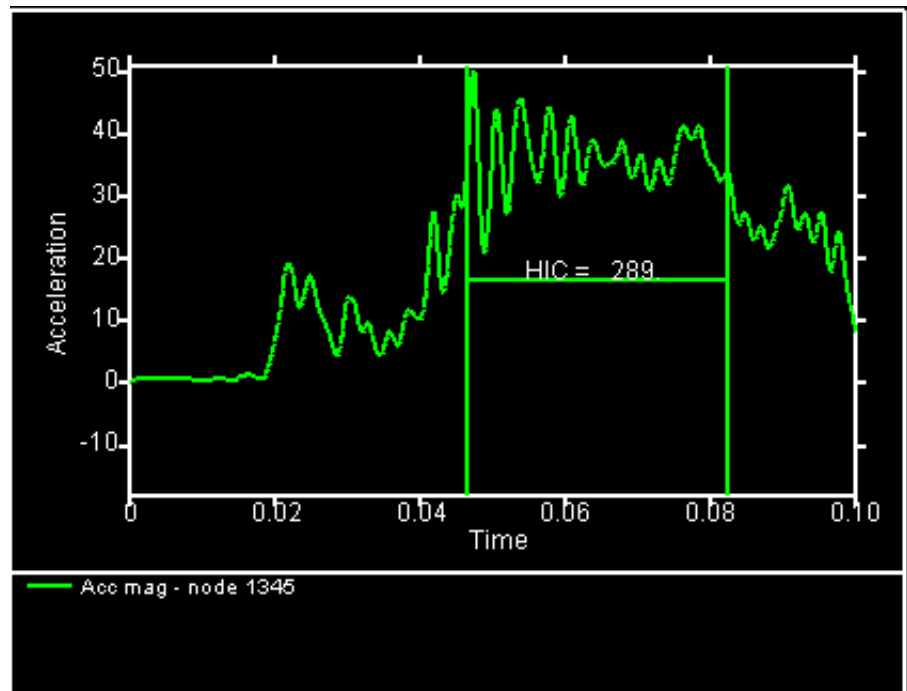
If the y-axis values are not in (G) then an optional factor can be specified that T/HIS will **DIVIDE** the y-axis values by to convert them to (G).

Example factors for different units are :

Unit	Factor
m/s <sup>2</sup>	9.81
mm/s <sup>2</sup>	9810
mm/ms <sup>2</sup>	0.00981

In addition to calculating and reporting the HIC value the time window and value can be displayed on the graph using the **Show HIC Value** option.

See [Appendix E](#) for more details on the Head Impact Criteria calculation.



### 5.11.9 HIC(d)

HIC(d) is used to calculate the Head Injury Criteria for the Free Motion Headform used within the FMVSS201 legislation. The equivalent dummy HIC(d) is calculated as follows

$$\text{HIC}(d) = 0.75446 \times (\text{free motion headform HIC}) + 166$$

### 5.11.10 CLI

Calculates the **3ms clip** value from an acceleration time history. Normally this option writes the value to the screen, and produces a curve of the clip region.

By default the screen value will be labeled as "**3ms** = value". This label can be modified by specifying a different **Screen Label**.

If required the values may also be written out to a file using the **WRITE TO FILE** option. In addition to calculating and reporting the 3ms clip value the time window and value can be displayed on the graph using the **Show 3ms Clip Value** option.

See [Appendix E](#) for more details on the 3ms clip calculation.

### 5.11.11 EXC

Calculate and displays an **EXC**eedence plot. This is a plot of force (y-axis) versus cumulative time (x-axis) for which the force level has been exceeded. By default the **Automatic** option will create an exceedence plot using either the +ve OR the -ve values depending on which the input curve contains most of. The **Positive** option will calculate the exceedence plot using only the points with +ve y values. The **Negative** option will calculate the exceedence plot using only the points with -ve y values.

### 5.11.12 VC

Calculates the **Viscous Criteria** from an acceleration time history. The user is prompted for the constants A and B. See [Appendix E](#) for more details on the VC calculation.

### 5.11.13 ASI

**Acceleration Severity Index.** This value is used to assess the performance of road side crash barriers.

This option requires 3 acceleration input curves. The user is prompted for the acceleration limits in the 3 directions.

The calculation method can be set to 2010 (BS EN 1317-1:2010) or 1998 (BS EN 1317-1:1998). See [Appendix E](#) for more details on this calculation.

### 5.11.14 THIV

**Theoretical Head Impact Velocity and the Post Impact Head Deceleration.** These values are used to assess the performance of road side crash barriers.

This option requires 3 input curves, a longitudinal and lateral acceleration and a rotation rate. The user is prompted for the constants Dx, Dy and Xo. See [Appendix E](#) for more details on these calculations.

### 5.11.15 NIJ

Biomechanical neck injury predictor. Used as a measure of injury due to the load transferred through the occipital condyles.

This option requires 3 input curves. 1 to represent Shear force, 1 to represent Axial force and a third to represent bending moment in the dummy's upper neck loadcell. Enter these curves in the corresponding input boxes.

The 4 critical constants used to calculate NIJ; Fzc (tension), Fzc (comp), Myc (flexion) and Myc (extension) default to the values specified by the test creators. These can be changed by entering different values into the respective boxes.

Enter the e distance into the e (distance) box.

Select which curves you wish to output to in the Output box.

For more information on the calculation of NIJ, refer to [Appendix E](#)

NIJ will output 4 curves due to the 4 possible loading conditions for NIJ;

Nte is the tension-extension condition

Ntf is the tension-flexion condition

Nce is the compression-extension condition

Ncf is the compression-flexion condition

**5.11.16 TTI**

Thorax Trauma Index:

This option requires 3 input curves. 1 to represent the Upper Rib Acceleration, 1 to represent the Lower Rib Acceleration and a third to represent the Lower Spine Acceleration. Enter these curves in the corresponding input boxes.

The output can either be written to the screen, appearing in a listing box, or written to a file specified in the File: input box, or both.

If the write to screen tab is highlighted, the following window will appear:

For more information on the calculation of TTI, refer to [Appendix E](#)

**5.11.17 NOR(y)**

Normalise the curve so that the Y values are within the range [-1, +1].

**5.11.18 NOR(x)**

Normalise the curve so that the X values are within the range [-1, +1].

**5.11.19 REG**

Make a curve have a constant time step.

It is necessary for a curve to have a constant time step between points for it to be filtered. This option takes an existing curve and prompts the user for a new time step. The points of the output curve are calculated by linear interpolation. Regularising a curve may alter its peak values, and could change filtered output slightly.

**5.11.20 VEC**

Calculate the vector magnitude of three input curves.

**5.11.21 VEC2D**

Calculate the vector magnitude of two input curves.

**5.11.22 ACU**

Airbag control Unit

### 5.11.23 **COR1**

Curve correlation function.

The Correlation function provides a measure of the degree to which two curves match. When comparing curves by eye, the quality of correlation may be judged on the basis of how well matched are the patterns of peaks, the overall shapes of the curves, etc, and can allow for differences of timing as well as magnitude. Thus a simple function based on the difference of Y-values (such as T/HIS ERR function) does not measure correlation in the same way as the human eye. The T/HIS correlation function attempts to include and quantify the more subtle ways in which the correlation of two curves may be judged.

The input parameters for the COR1 function have been chosen so as to produce a strict judgement of the correlation (see Appendix F for more details).

### 5.11.24 **COR2**

The COR2 function is the same as COR1 except the input parameters have been chosen so as to produce a less strict judgement of the correlation (see Appendix F for more details).

### 5.11.25 **COR3**

Another curve correlation function.

This function first normalises the curves using two factors either specified by the user or defaults calculated by the program (the maximum absolute X and Y values of both graphs). For each point on the first normalised curve, the shortest distance to the second normalised curve is calculated. The root mean square value of all these distances is subtracted from 1 and then multiplied by 100 to get an index between 0 and 100. The process is repeated along the second curve and the two indices are averaged to get a final index. The higher the index the closer the correlation between the two curves.

Note that the choice of normalising factors is important. Incorrect factors may lead to a correlation index outside the range of 0 to 100 (see Appendix F for more details).

### 5.11.26 **WIF**

Weighted Integrated Factor (WIFAC) curve correlation function.

Compares curves using the Weighted Integrated Factor method (WIFAC). A value between 0 and 100 is calculated, the higher the index the closer the correlation between the two curves.

See Appendix F for more details.

## 5.12 SEISMIC Options

The **SEISMIC** menu is shown in the figure (right). T/HIS can be used to handle response spectra information. In particular, displacement, velocity or acceleration spectra can be read and converted to another format.



- 5.12.1 DV** Displacement spectrum is converted to a velocity spectrum
- 5.12.2 DA** Displacement spectrum is converted to an acceleration spectrum
- 5.12.3 VD** Velocity spectrum is converted to a displacement spectrum
- 5.12.4 VA** Velocity spectrum is converted to an acceleration spectrum
- 5.12.5 AD** Acceleration spectrum is converted to a displacement spectrum
- 5.12.6 AV** Acceleration spectrum is converted to a velocity spectrum.
- 5.12.7 DS** Produce a design spectrum from a response spectrum through the specification of a broadening factor..
- 5.12.8 RS** Produce a response spectrum from input accelerations. This gives the response of a damped single degree of freedom system, given its damping factor and period, to the input acceleration time-history.
- 5.12.9 FFT** Perform a fast Fourier transform. Convert an input signal from the time to the frequency domain.

There are three options for output;

- magnitude only
- magnitude and phase
- real and imaginary components of the time signal.

The frequency is calculated in Hz NOT radians/s if the time axis is in seconds.

T/HIS automatically adds points with zero y-value to the end of the curve to pad the curve out so that the number of points is increased to the next power of 2.

There are two options for scaling the curves output:

- Scaling Option 1 - Consistent with other signal processing software giving a magnitude independent of any padding. This is the default and recommended for most purposes. Performing an inverse FFT on the resulting curves will NOT get back exactly to the original curve if it did not have a number of points equal to a power of 2.
- Scaling Option 2 - With this option, applying an inverse FFT to the resulting curves will generate a curve the same as the original even if the original curve did not have a number of points equal to a power of 2. This is useful if users wish to create their own filters, where the filter characteristic is defined in the frequency domain.

An option to regularise the curve before performing the function is on by default. The spacing between points on the frequency axis of the resulting curve is determined by the time duration of the padded input curve;  $dx = 1.0/(\text{time})$ .

The highest frequency in the output curve is determined by the time interval of the input curve;  $F(\text{max}). = (\text{\#points})/dt$



- 
- 5.12.10 IFFT** Performs an inverse fast Fourier transform. Converts two input signals from the frequency to the time domain. The two input signals can be the magnitude and phase or real and imaginary components of the time signal.
- NOTE: If an FFT using scaling option 1 is performed on a curve that does not have a number of points equal to a power of 2 and then an IFFT is performed on the resulting curves you will NOT get back exactly to the original curve. This is because the FFT and IFFT both scale their output curves by the number of points in the curve, which in this case will be different. For the FFT the number of points used to scale the curves is the original number of points before padding. For the IFFT the number of points used is the original number of points plus the points needed to make it a power of 2.
- If the number of points in the original curve is a power of 2 and no padding is required, the IFFT of the resulting curves will get back to the original curve.
- 5.12.11 NCP** By default beam element plastic rotations are always written out by LS-DYNA as being increasing +ve (i.e. cumulative). This option allows a non-cumulative plastic rotation to be calculated by taking two input curves: the moment/time and the cumulative rotation/time histories for the beam in question.
- 5.12.12 BLC** Baseline correction.

## 5.13 MACRO Options

The **MACRO** menu can be used to play FAST-TCF based macro files on existing T/HIS curves.

T/HIS macro files can be stored in any directory. Each user can define up to three macro areas using the oa\_pref option:

**this\*macro\_directory:**  
C:\blah\macros

T/HIS will read any **.thm** files within the macro directories and generate the macro menu (shown right) using keywords within the macro scripts. T/HIS will search the macro directories in the following order.

this\*macro\_directory from SYSTEM oa\_pref file  
this\*macro\_directory from users oa\_pref file  
this\*macro\_directory from local oa\_pref file

The scripting for a macro in T/HIS is based upon FAST-TCF using variables within the script (see [FAST-TCF section](#) for more details). There are some extra command options to make T/HIS aware of input curves and constants within the macro. These macro keywords are as follows:

macro acr <name>

acronym for the macro button in T/HIS

macro title <description>

a more descriptive title for the macro

macro curve <curve variable> <curve description>

FAST-TCF variable name for input curve followed by curve description

macro const <const variable> <curve description>

FAST-TCF variable name for input constant followed by constant description

If one or more macro files are found with duplicate acronyms then only the last file read will be displayed so users can override SYSTEM macros with there own definitions if they want to.

When the user selects one of the Macro functions the macro file associated with the function is read and T/HIS creates a selection menu for the user to define the relevant curve numbers and constant values to input into the macro script. These inputs will **replace** the variables used within the macro.

For example: If the user selects **#1** for the macro curve variable **macro\_input**, then any occurrence of **\$macro\_input** in the macro script will be replaced by **#1**.

An example macro script follows. This macro asks the user for a filter option (e.g. c60, c600, c1000) and also an input curve number. The macro then filters the input curve and divides by 9810.

```
# Macro to convert a file to g after filtering
#
macro acr to_g
macro title filter and convert curve to g
macro curve macro_input input curve
macro const macro_filter filter to use
#
model none
model 1
oper $macro_filter $macro_input tag filtered
oper div filtered 9810.0
```

## 5.14 FAST-TCF Options

The **FAST-TCF** menu can be used to capture and playback FAST-TCF scripts. FAST-TCF is a simple and intuitive scripting language for T/HIS. See [FAST-TCF \(section 7\)](#) for more details and commands.

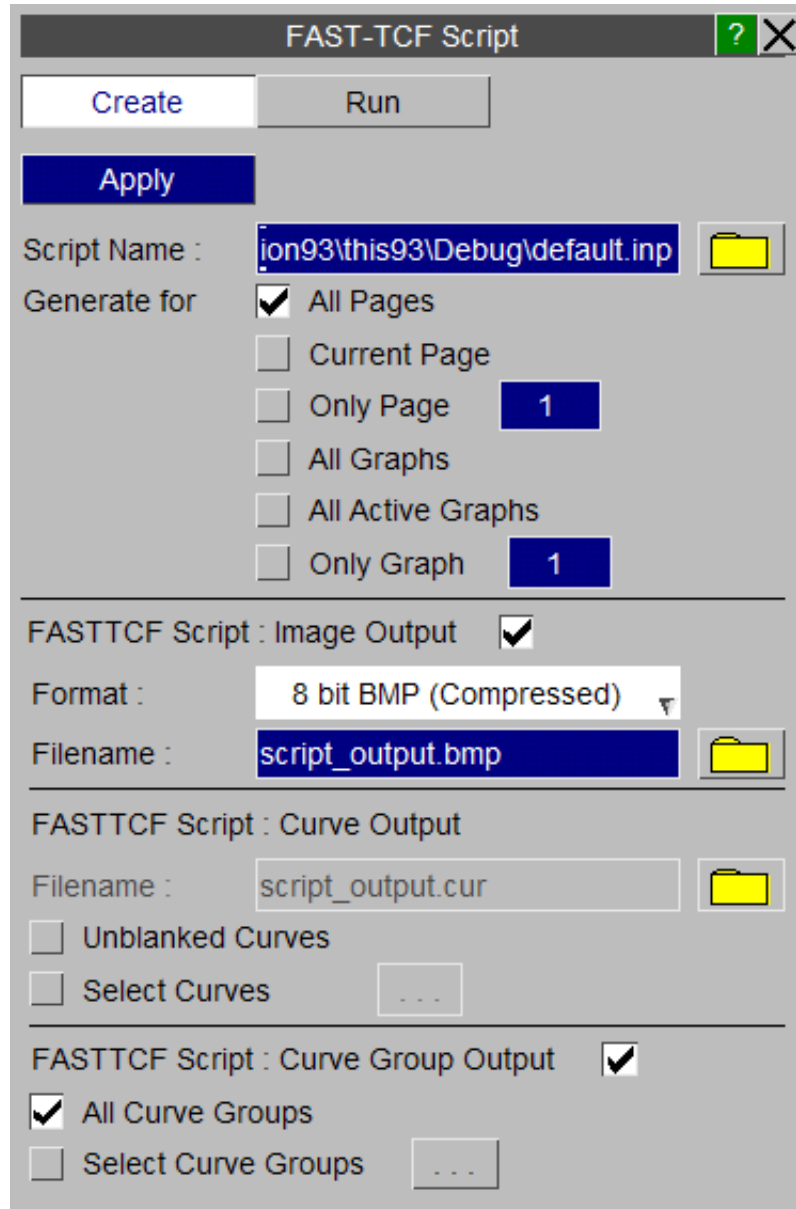
### 5.14.1 Create

T/HIS 9.2 onwards has the ability to automatically generate FAST-TCF scripts using the CREATE menu.

The FAST-TCF script will contain all of the commands required to

1. Create and position multiple graphs on pages.
2. Extract the data from models or other files
3. Carry out any curve operations required to reproduce the chosen curves
4. Set any curve styles and labels
5. Set plot attributes such as titles, axis labels, colours, fonts and scaling
6. Generate the output image and/or curve file
7. Generate curve groups

Before generating the FAST-TCF script the following options can be set



**FAST-TCF Script** [?] [X]

**Create** **Run**

**Apply**

Script Name :  [Folder Icon]

Generate for

- ☒ All Pages
- ☐ Current Page
- ☐ Only Page
- ☐ All Graphs
- ☐ All Active Graphs
- ☐ Only Graph

---

FASTTCF Script : Image Output ☒

Format :  [Dropdown Arrow]

Filename :  [Folder Icon]

---

FASTTCF Script : Curve Output

Filename :  [Folder Icon]

- ☐ Unblanked Curves
- ☐ Select Curves

---

FASTTCF Script : Curve Group Output ☒

- ☒ All Curve Groups
- ☐ Select Curve Groups

## Generate For

### All Pages

The FAST-TCF script will contain all of the commands required to regenerate all of the pages that contain 1 or more graphs.

If the option to generate images is selected then the FAST-TCF script will contain the commands to generate multiple images with the page number appended to the filename specified.

### Current Page

The FAST-TCF script will contain all of the commands required to regenerate the currently displayed page.

### Only Page (n)

The FAST-TCF script will contain all of the commands required to regenerate the selected page.

### All Graphs

The FAST-TCF script will contain all of the commands required to regenerate all the currently defined graphs.

All of the graphs will be positioned on page 1 using the currently defined layout.

*This option will only be available if T/HIS only contains a single page ([see Section 3.2](#)).*

### All Active Graphs

The FAST-TCF script will contain all of the commands required to regenerate all of the active graphs.

All of the graphs will be positioned on page 1 using the currently defined layout.

*This option will only be available if T/HIS only contains a single page ([see Section 3.2](#)).*

### Only Graph (n)

The FAST-TCF script will contain all of the commands required to regenerate the selected graph.

The graph will be positioned on page 1.

## FAST-TCF Script : Image Output

If this option is selected then the FAST-TCF script will contain the commands required to generate an image of each of the pages/graphs selected for output. The **Image Format** can be set to any of the supported image types ([see Section 5.8](#)).

If the FAST-TCF script generates multiple pages then the **Filename** specified will be used for the first image. Subsequent images will use the specified filename with "\_2", "\_3" ... appended.

## FAST-TCF Script : Curve Output

By default the FAST-TCF script will only contain the command needed to reproduce the curves that are unblanked in 1 or more of the graphs selected for output. This option can be used to select additional curves for which the commands required to generate them are also added to the FAST-TCF script. If a curve is selected that is also unblanked in one of the graphs the command to regenerate it are only added to the FAST-TCF script once.

In addition to selecting additional curves this option can also be used to add commands to the FAST-TCF script to write the additional curves out to a T/HIS .cur curve file.

## FAST-TCF Script : Curve Group Output

This option can be used to select additional curves for output to the FAST-TCF script by curve group. If a curve is selected that is also unblanked in one of the graphs the command to regenerate it are only added to the FAST-TCF script once. This option will also add the commands to regenerate the selected curve groups to the FAST-TCF script.

## 5.14.2 Run

This menu allows the user to run a FAST-TCF file from within T/HIS. After the user has selected the FAST-TCF file T/HIS scans the file for data requests and model requests to see what input the FAST-TCF file requires. Note that there must be a model read into T/HIS before a FAST-TCF file that contains data extraction can be run.

The next FAST-TCF command line is displayed in red in the upper text area, at this point the user can select to **Play** the FAST-TCF file or **Step** through it line by line. After every line of FAST-TCF the resulting command in T/HIS is shown in the lower text area. Select **End** during stepping through the lines to go to the end of the file. **Reread** will re read the file and start back at the beginning.

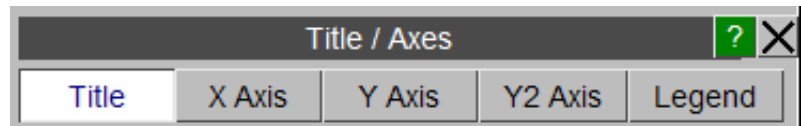
The **Model Mapping** option allows the user to define which model in T/HIS should be used for the equivalent model number in the FAST-TCF script. The model number **zero** is equivalent to the default model in FAST-TCF if no models are defined. The default model mapping will use the same model numbers as in the FAST-TCF script.

**Auto confirm text boxes** will force T/HIS to confirm any text boxes that should appear in the interactive playback of a FAST-TCF script (such as HIC results and so on).

The FAST-TCF script will ignore any existing T/HIS curves and their tags. This guarantees that the user can run a single FAST-TCF file many times and it will only use the new curves created by FAST-TCF.



## 5.15 TITLE/AXES/LEGEND Options



The **TITLE/AXES** menu is shown in the figure (right).

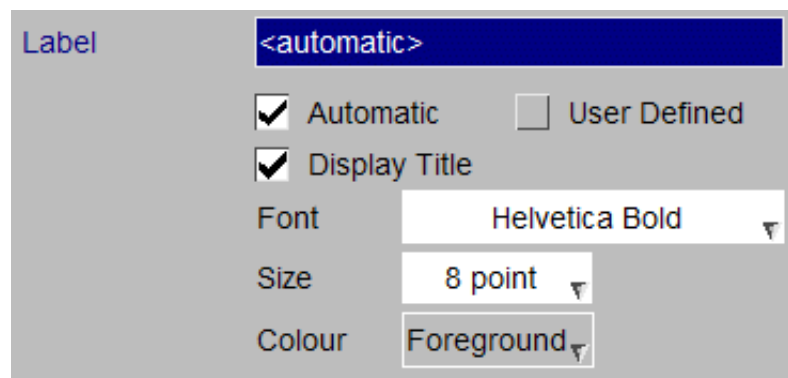
This menu controls the contents of the title and axes labels and the axis scaling.

The individual axis, title and legend menus can also be accessed by clicking over the appropriately highlighted area on the graph.

Changes to the TITLE/AXES/LEGEND options are only applied to active graphs ([see Section 3.5](#)).

### 5.15.1 TITLE

The plot title may be set **AUTO**matically or manually. When the **AUTO** option is selected the text box will display **<automatic>** and the plot title will be set to the title of the first curve that is currently being plotted. The plot title may be turned on and off by toggling the **ON/OFF** button.



## 5.15.2 X-AXIS

### AXIS LABELS

The x-axis label may be set automatically or a user defined label can be specified.. When the **AUTOMATIC** option is selected the text box will display **<X automatic>** and the axis label will be set to the x axis label of the first curve that is currently being plotted. The axis label may be turned on and off by toggling the **Display Label** button.

In addition to displaying the axis labels an optional unit label can also be appended to the axis label. If the option to add a unit label is set to Automatic then the unit label displayed will depend on the current curves that are visible and the current unit system being used to display results (see [Section 5.22](#) for more information on Unit Systems). If the curves being displayed do not have the same axis unit then no unit label will be displayed. The unit label may be turned on and off by toggling the **Add Units** button.

### AXIS LIMITS

The minimum and maximum x axis values can be controlled using a combination of the text box and the popup menu opposite.

#### Autoscale

The axis values will be set to the maximum and minimum values of all the curves that are currently being plotted.

#### Locked

The axis limit is set to the user defined value specified in the text box. If the curves are translated or scaled dynamically the limit will be reset.

**Title / Axes**

**Label**

☒ Automatic ☐ User Defined

☒ Display Label

☒ Add Units **<automatic>**

☒ Automatic ☐ User Defined

Font **Default**

Size **Automatic**

Colour **Foreground**

**Minimum** **<auto>** ☒ Autoscale ☐ Locked

**Maximum** **<auto>** ☒ Autoscale ☐ Locked

**Axis Type** ☒ Linear ☐ Logarithmic

**Grid Spacing** ☒ Automatic ☐ User Defined

**Interval** **0.0000E+00**

**Offset** **0.0000E+00**

**Units** ☒ Add Exponent to Label

Format **Automatic**

Decimal Places **3**

Font **Default**

Size **Automatic**

Colour **Foreground**

Note : The global command **AUTOSCALE** (see [Section 4.5](#)) will reset the minimum and maximum values to **AUTO**.

### AXIS TYPE

The x-axis can be switched between a **Linear** or **Logarithmic** scale. If a **Logarithmic** scale is selected a warning will be generated if an attempt is made to plot points that have -ve or zero X values and the points will be skipped.

### GRID SPACING

By default T/HIS will automatically set the grid line intervals for the x-axis when the grid is turned on ( see [Section 5.16.6](#) ). If the GRID option is changed from **Automatic** to **Manual** a grid **Interval** and **Offset** may be specified. If the **Interval** is set to 0.1 and the **Offset** to 0.02 then grid lines will be produced at 0.02, 0.12, 0.22 ....



# UNITS

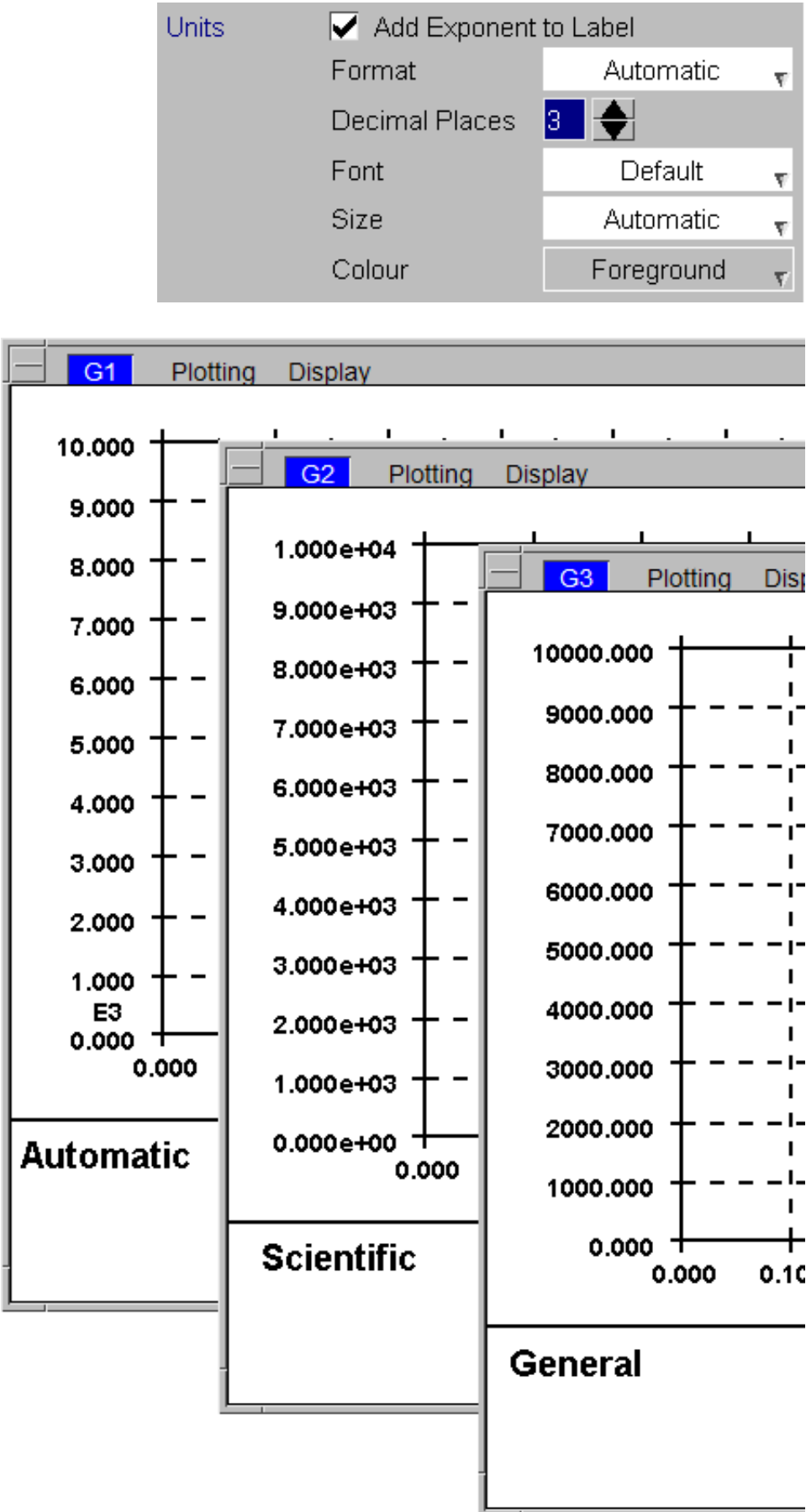
Axis values can be displayed using 3 different formats

**Automatic**  
Values are displayed using exponential format, all values are displayed as values of E0, E3, E6 etc.  
e.g 11.234E+03

**Scientific**  
Values are displayed using exponential format.  
e.g 1.123E+04

**General**  
Values are displayed as real numbers.  
e.g 11234.000

In addition to specifying the format, the number of decimal places can also be set between 0 and 9 and the colour and font used to display the values can be set.



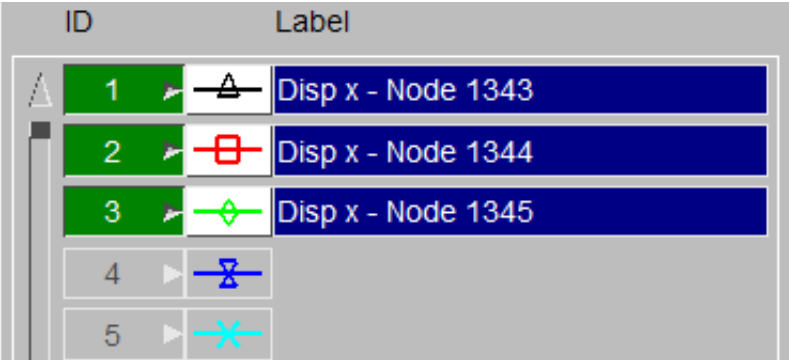
### 5.15.3 Y-AXIS

The same options for LABELS, LIMITS, SCALE, GRID LINES and UNITS apply to the Y-AXIS as those available for the X\_AXIS.

### 5.15.4 Second Y-AXIS

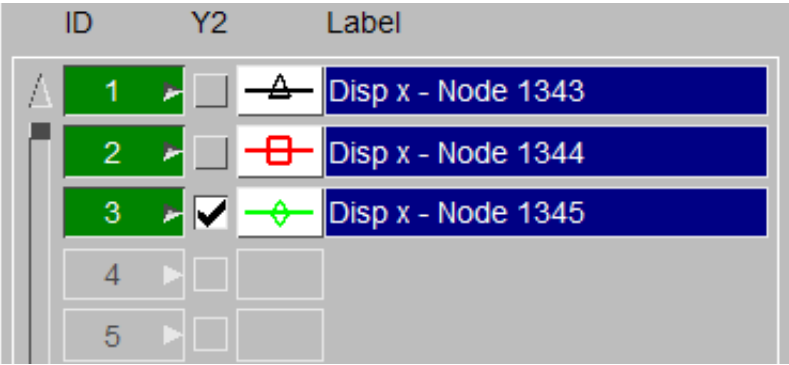
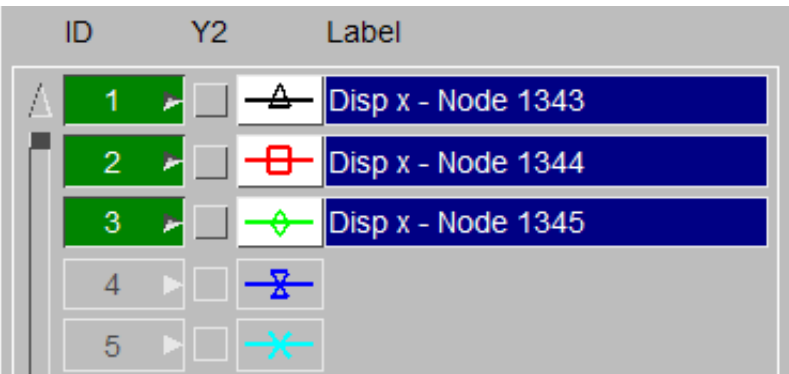
Curves can be plotted in T/HIS using 2 different y-axis scales. When **DOUBLE Y-AXIS** is selected using the check box in the Y2 Axis menu the curve management window changes

from



to

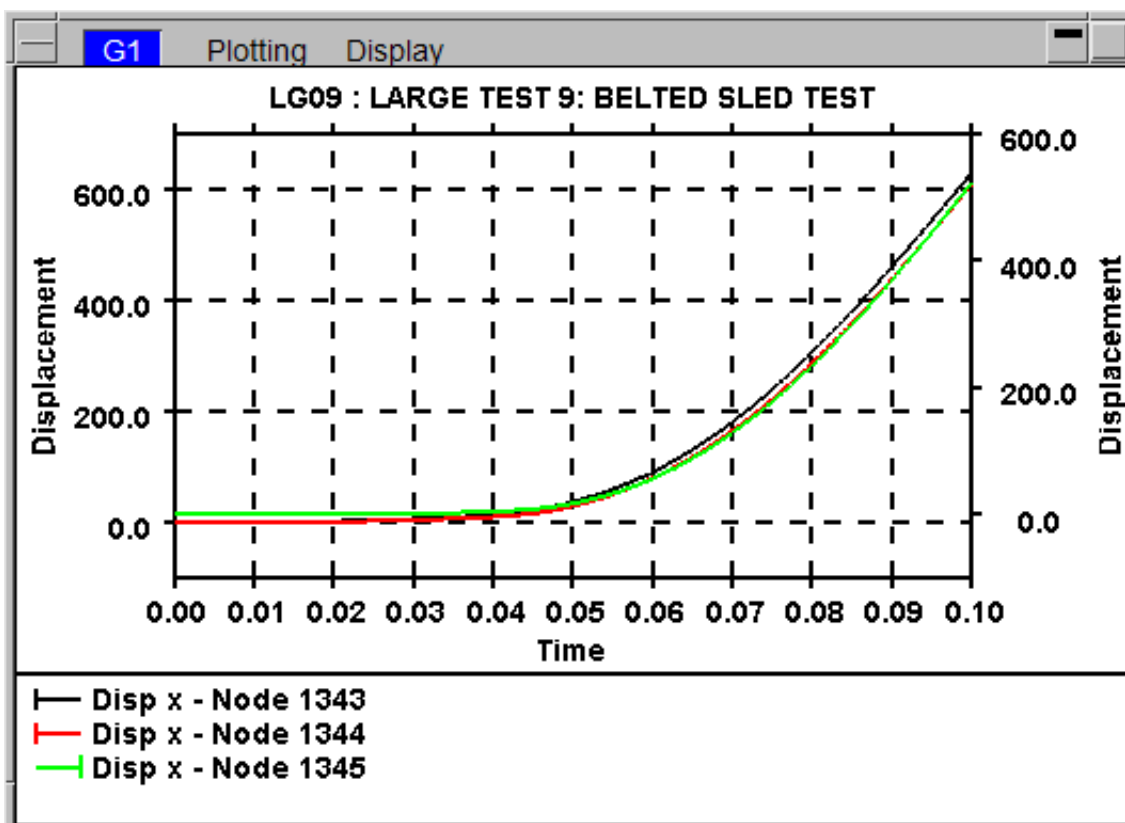
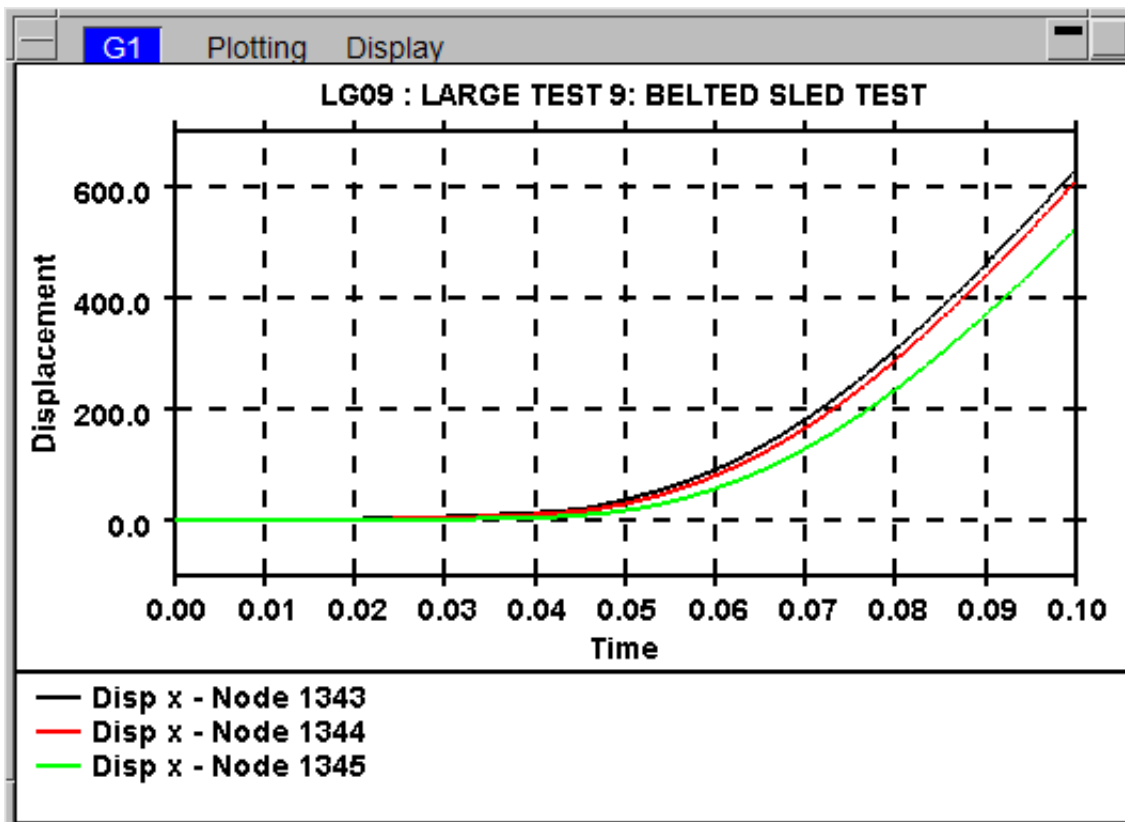
with an additional tick box for each curve that controls which curves are plotted against the second (right hand) y-axis.



If only one y-axis scale is used it is not possible to meaningfully plot curves with different units or very different values. A second scale allows more information to be displayed at once, as demonstrated below.

To identify which axis a curve is being plotted against the line labels on the plot are automatically modified.

Second Y axis disabled	— Disp x - Node 1343
Left hand Y axis	└─ Disp x - Node 1343
Right hand y axis	─┤ Disp x - Node 1343



All of the options that are available to control the label, scale and type of the y-axis are also available for the second y-axis except for the Grid option.

NOTE : When the DOUBLE AXIS option is used with GRID lines a grid is only plotted for the left hand y-axis.

## 5.15.5 Legend

### 5.15.5.1 Curve Labels

#### Show Prefix

This option can be used to automatically add a prefix to each of the curve legends when a curve is plotted. This option has 3 settings

##### Automatic

If there is more than 1 model loaded in T/HIS then a prefix will automatically be added to any curves that have been read in from a model. Curves read in from other files will not be prefixed.

##### On

A prefix will automatically be added to any curves that have been read in from a model regardless of the number of models currently loaded in T/HIS. Curves read in from other files will not be prefixed.

##### Off

No prefixes will be added

#### Prefix Format

This option can be used to set the format used for the curve prefix. This option has 4 settings

<b>Model Number</b>	The model number will be used as the prefix. e.g <b>(M1)</b>
<b>Directory</b>	The directory name the model was read from will be used at the prefix. e.g. <b>(/run1)</b>
<b>THF File</b>	The root name of the THF file will be used as the prefix. e.g <b>(sled_test)</b>
<b>User Defined</b>	A user defined prefix will be used. The prefix can be defined on a model by model case using the <a href="#">Model Menu</a> .

The font, size and colour of the text used to display the legends can also be specified.

The screenshot shows the 'Legend' tab of the 'Title / Axes' dialog. The 'Curve Labels' section has 'Show Prefix' set to 'Automatic', 'Prefix Format' to 'Model Number', 'Font' to 'Default', 'Size' to 'automatic', and 'Colour' to 'Foreground'. The 'Layout' section has 'Column List' unchecked, 'Auto' checked, 'Off' unchecked, 'Floating' unchecked, '1 Column' unchecked, '2 Columns' checked, and '3 Columns' unchecked. The 'Background' section has 'Colour' set to black and 'Transparency' set to 0. The 'User Lines' section has 'Display Lines' checked, 'Size' set to 'automatic', 'Font' set to 'Default', and 'Colour' set to 'Foreground'. A 'Reset' button is located below the 'User Lines' section.

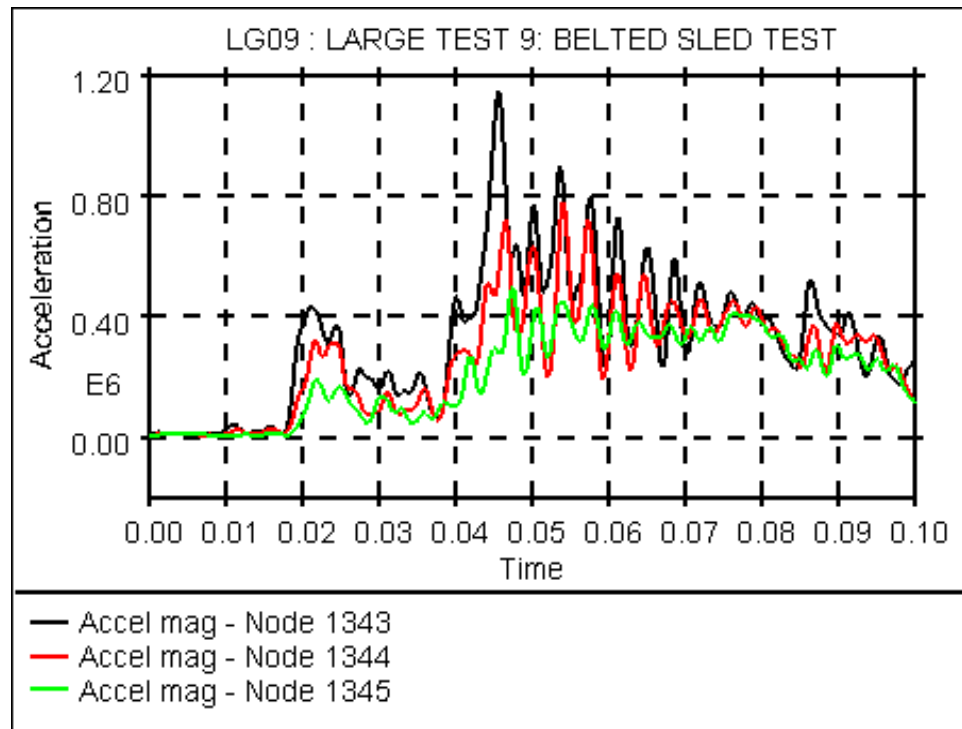
### 5.15.5.2 Layout

T/HIS has 4 different plotting formats as described below. The number of columns used to display the curve legends can also be set between 1 & 3. When multiple columns are used curve labels will automatically be truncated to fit the available space.

#### Column List (default)

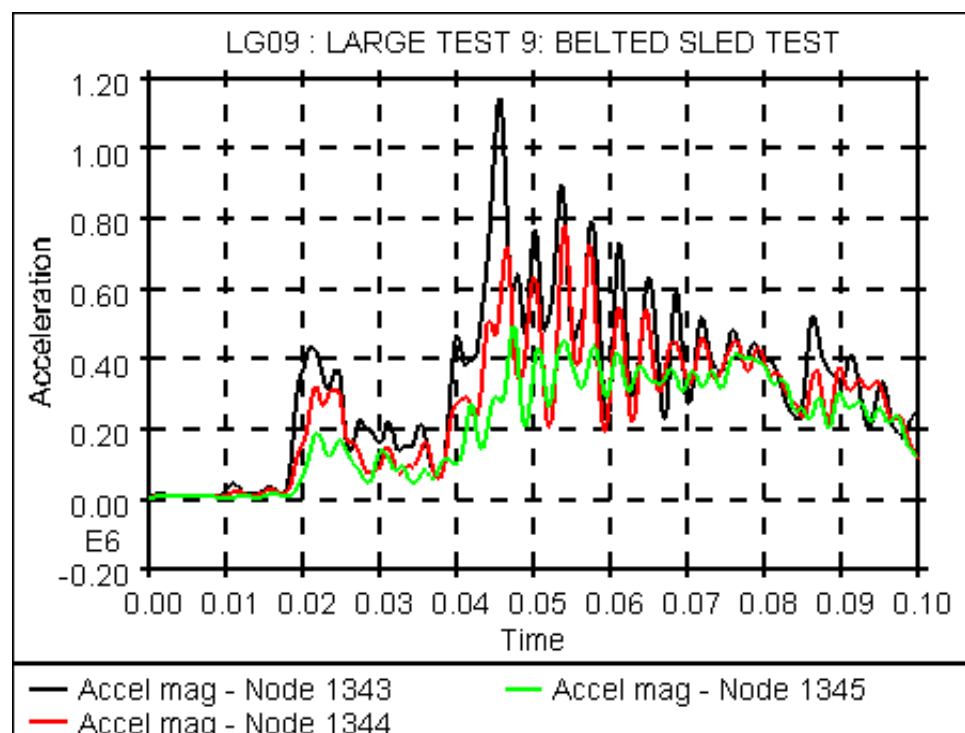
This format fixes the size of the plotting area. The maximum number of curve legends that can be displayed will depend on the font family and size selected by the user and the number of columns.

If any [USER LINES](#) have been defined then the area used to display the legend will be reduced so that the text does not cover the



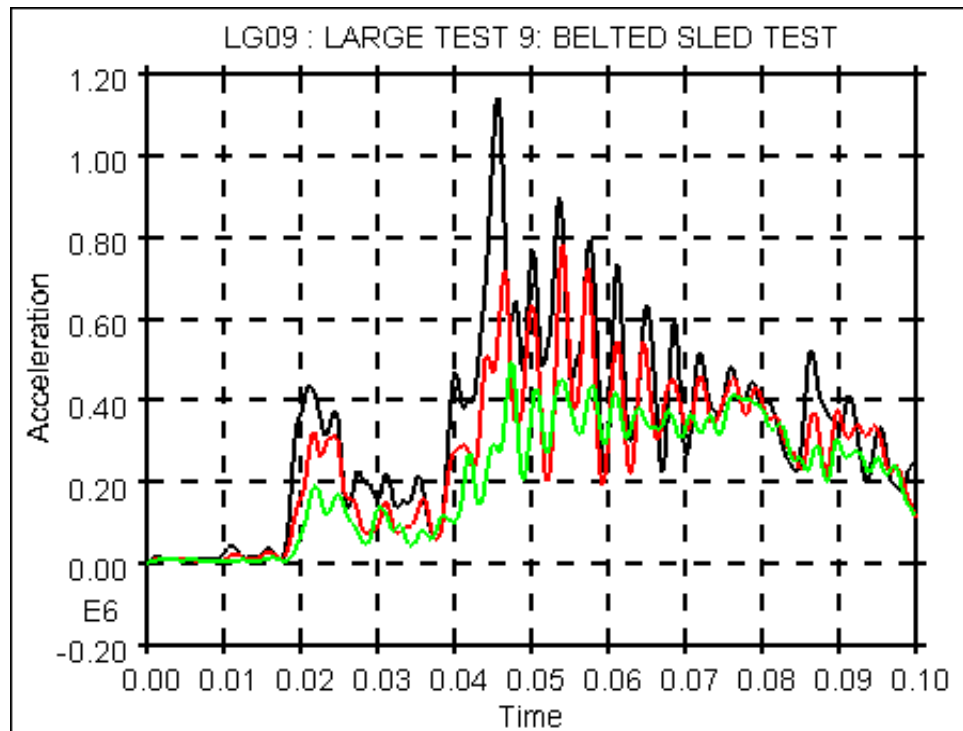
#### Automatic

This format automatically adjusts the plot size to maximise the plotting area while still showing a maximum of 18 line labels. Any text entered using the [USER LINES](#) option will be ignored in this plotting mode.

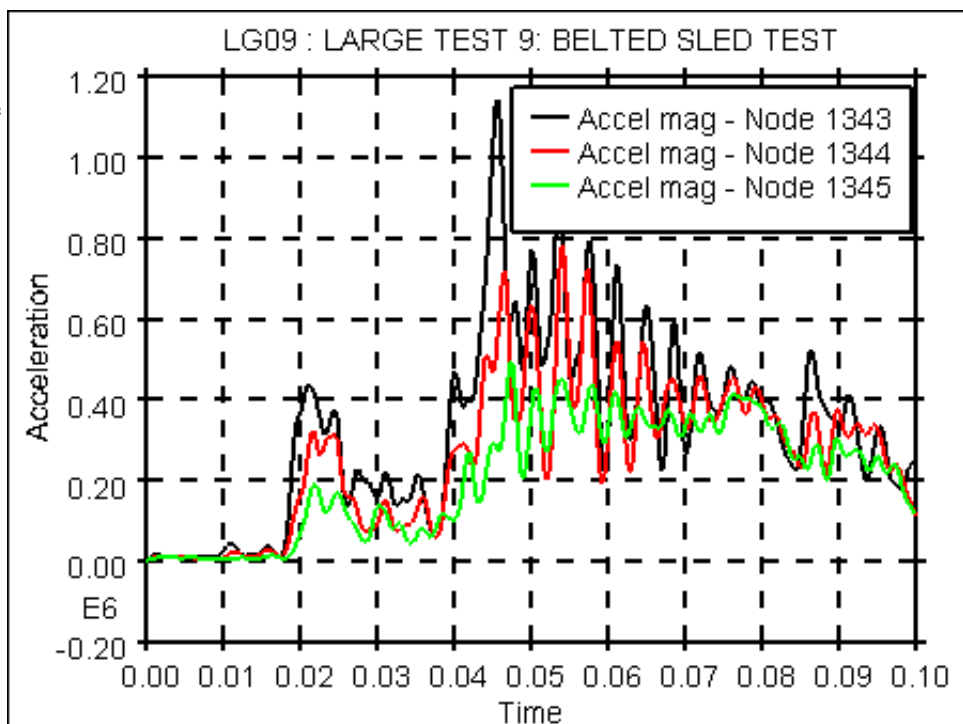


**Off**

This format turns **OFF** the display of the graph legend and maximises the plotting area by not showing any line labels. Any text entered using the **USER LINES** option will be ignored in this plotting mode.

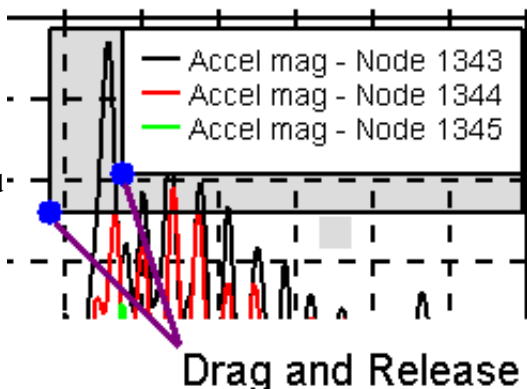
**Floating**

This format maximises the plotting area and positions the legend on top of the graph area.



The size of the legend can be modified by clicking with the left mouse button on the legend border/corner and dragging.

The legend can also be moved by clicking with the left mouse button inside the legend and dragging.



### 5.15.5.3 BACKGROUND

This option can be used to alter the default background colour of the floating legend. By default the colour will be the same as the background colour of the graph. As well as setting a different background colour for the floating legend a %age transparency can also be specified if the legend obscures any curves

### 5.15.5.4 USER LINES

This option can be used to alter the default text that appears on the bottom right-hand corner of each plot. Text can be typed into any of the panels or they can be left blank. The **Size** of the text may be altered. If no text is specified the area used by the curve legends will be increased.

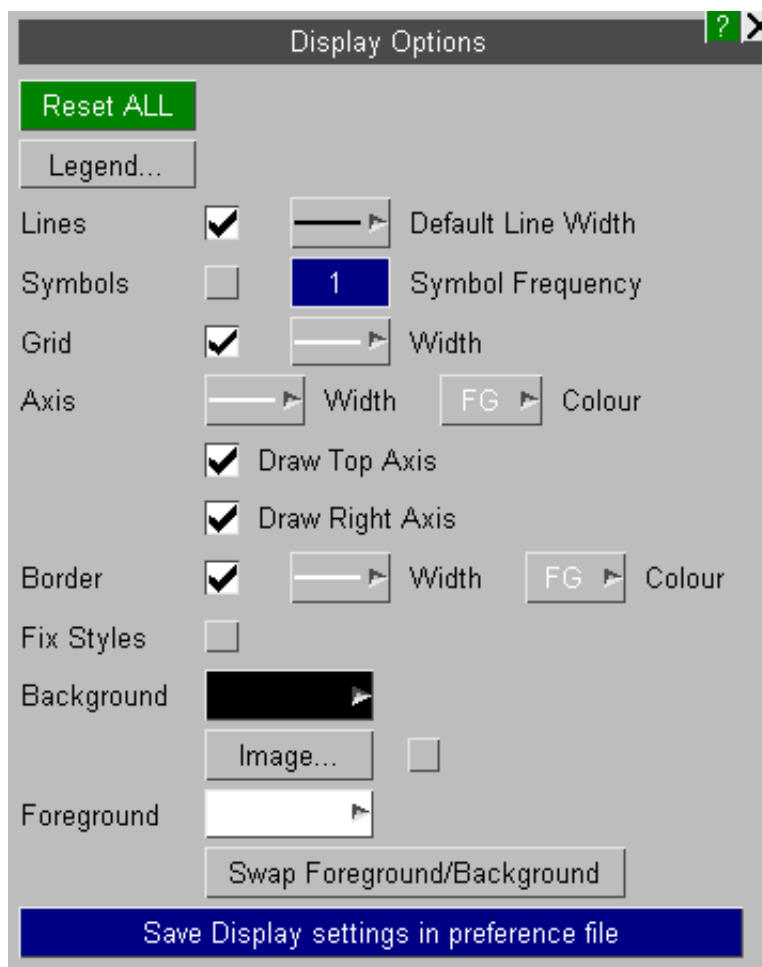
The default values are read from the "preferences" file (see [Appendix H](#) for more details).

## 5.16 DISPLAY Options

The **DISPLAY** menu is shown in the adjoining figure. This menu controls the overall appearance of plots.

As well as controlling basic things like the background colour and whether a grid is drawn this menu also controls a number of default settings that are applied to all curves. These default settings may be overwritten for individual curves using the **CURVE CONTROL** menu ([see Section 5](#))

Changes to the Display options are only applied to active graphs ([see Section 3.5](#))



### 5.16.1 LEGEND...

This option will map the Legend settings panel ([see Section 5.25.5](#))

### 5.16.2 LINES

Lines ☒ Default Line Width

This is an **ON/OFF** switch for the lines between points to be drawn for all curves. The default is **ON**. The **Default Line Width** is used for all curves that have not had their widths explicitly set in the **CURVE CONTROL** menu.

The default line width can be specified in the "preferences" file ([see Appendix H](#) for more details).

### 5.16.3 SYMBOLS

Symbols ☐ 1 Symbol Frequency

This is an **ON/OFF** switch which controls whether symbols are plotted on top of the curves to help identify them. This option affects all the curves that are currently being used. If you wish to turn the symbols on for only some of the curves then this switch should be set to **ON** and the **CURVE CONTROL** menu should be used to turn the symbols off on the curves for which you do not want symbols drawn on. The default is **OFF**.

The **Symbols Frequency** is used for all curves that have not had a frequency explicitly set in the **CURVE CONTROL** menu. This value controls how often a symbol is drawn on a curve.



### 5.16.3 GRID



This is an **ON/OFF** switch which determines whether or not grid lines are shown on the plot. The default is **OFF**. The **Grid Width** can be used to change the width of the grid and axis lines.

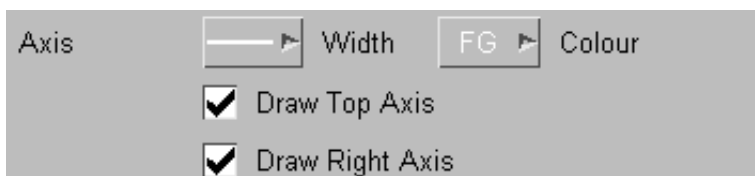
`/de grid on` turns grid lines on

`/de grid off` turns grid lines off

`/de grid th 2` sets the grid thickness to 2 pixels

The default grid width and visibility can be specified in the "preferences" file (see [Appendix H](#) for more details).

### 5.16.5 AXIS



The **Axis Width** can be used to change the width of the axis lines. The **COLOUR** button can be used to change the colour of the axis lines (see Section [5.6.2](#) for details on the available colours).

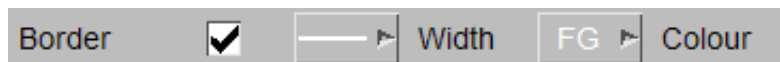
**Draw Top Axis** This option can be used to turn on and off the display of the graphs top axis

**Draw Right Axis** This option can be used to turn on and off the display of the graphs right hand axis

The default axis width can be specified in the "preferences" file (see [Appendix H](#) for more details).

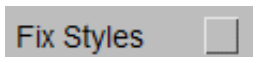
The default settings for these 2 options can also be specified in the "preferences" file (see [Appendix H](#) for more details).

### 5.16.6 BORDER



This is an **ON/OFF** switch which determines whether or not a border is drawn round the plot. The default is **ON**. The **Border Width** can be used to change the width of the border. The **COLOUR** button can be used to change the colour of the border (see Section [5.6.2](#) for details on the available colours).

### 5.16.7 FIX LINE STYLES



This is an **ON/OFF** switch which resets the curve styles when they are plotted on the screen so that the curves cycle through the default T/HIS colours and styles as they are plotted. This will result in the first curve being plotted always being white, the second red, the third green .etc regardless of their curve numbers. The default is **OFF**.

### 5.16.8 Background

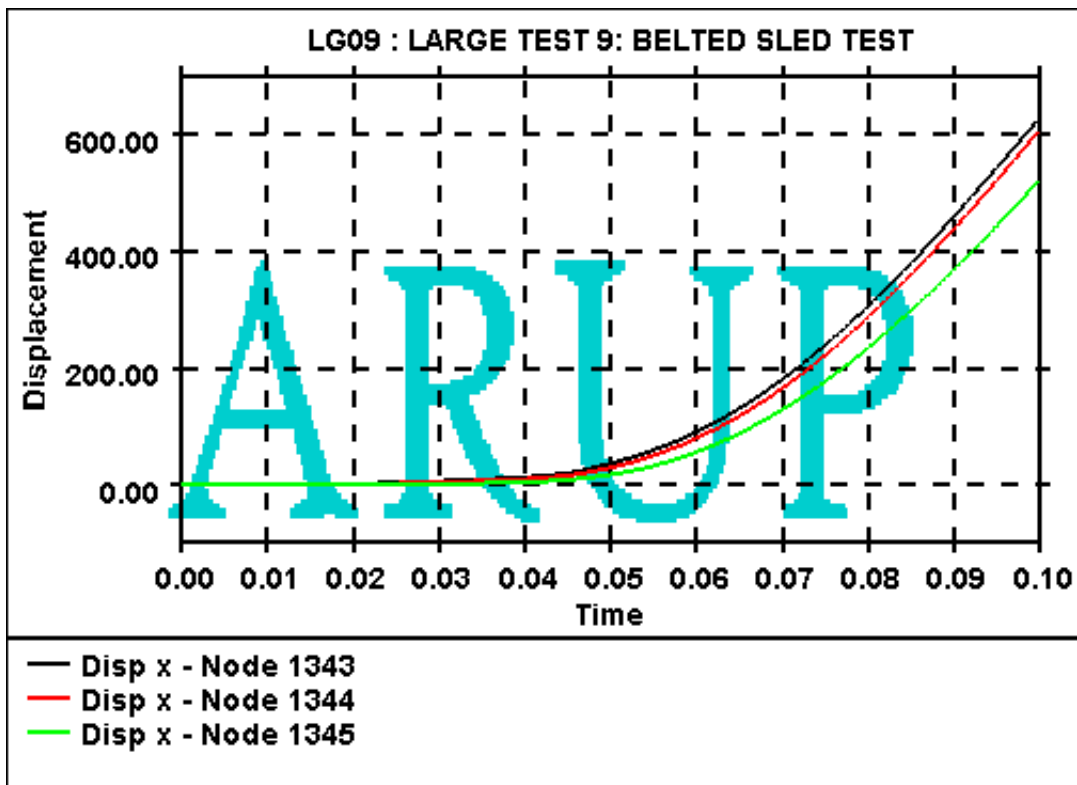
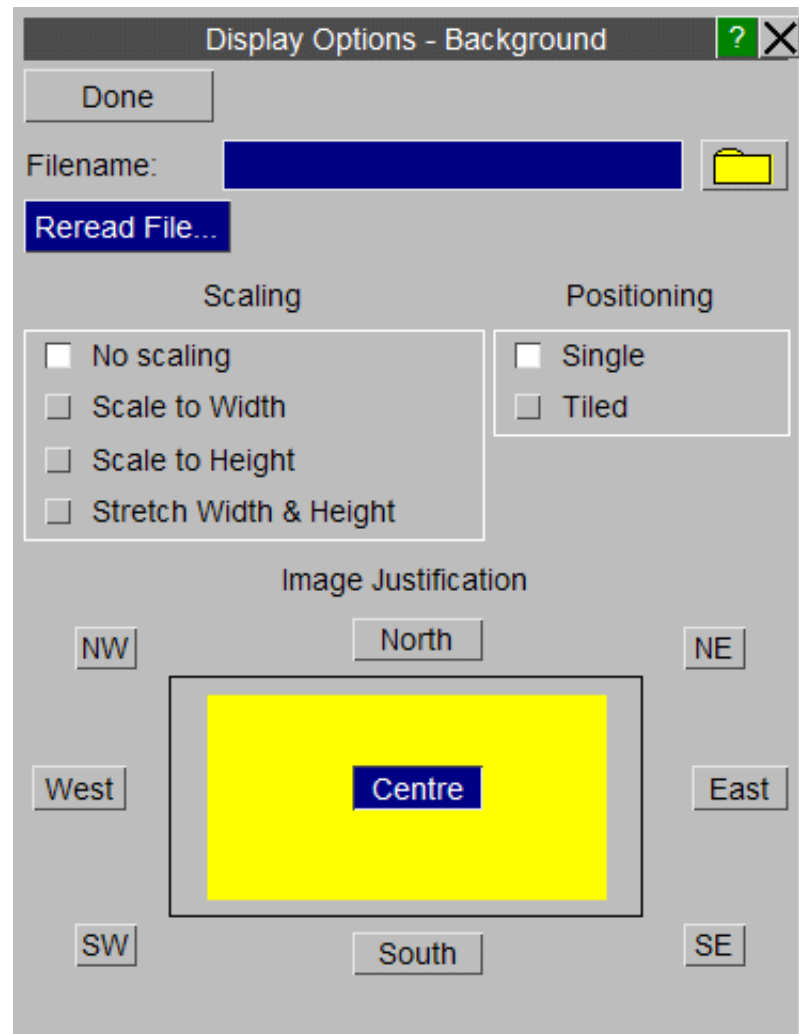


This option can be used to modify the background colour (see Section [5.6.2](#) for details on the available colours) or to set a background image. By default the background colour is set to BLACK.

## Image

The IMAGE option can be used to display a background image behind a graph instead of a solid background colour.

If the image dimensions do not match the graph window dimensions then the image can be scaled to fit or it can be tiled.



### 5.16.9 Foreground



This option can be used to modify the foreground colour (see Section [5.6.2](#) for details on the available colours). By default the background colour is set to BLACK and the foreground colour is set to WHITE.

Initially the grid, axes, border and labels are all set to the foreground colour.

### 5.16.10 Swap Foreground/Background

A screenshot of a button with the text 'Swap Foreground/Background' in a blue, sans-serif font. The button has a light grey background and a thin black border.

This option can be used to swap the currently defined foreground and background colours.

### 5.16.11 Display Max/Min

In versions of T/HIS prior to 9.4 the display of minimum and maximum curve values was controlled in the **DISPLAY** menu . In version 9.4 these options have been moved to the **PROPERTIES** menu (see section 5.20.21).

## 5.17 SETTINGS

### 5.17.1 Data Sources

This menu allows the user to specify their preferred order of data sources for the different data types. Upon reading in models T/HIS will read all files regardless of these preferences. When T/HIS extracts data for plotting the source is dependent on that currently set in this menu. If you select a data component or entity that is not available in the first data source T/HIS will automatically try the other data sources in order until the combination is found.

The [Model Manager](#) can be used to see what source has been used for each item for models already read into T/HIS

Settings			
Data	Files	General	Layout
	1st	2nd	3rd
Global	LSDA ▶	ASCII ▶	THF ▶
Parts	LSDA ▶	ASCII ▶	THF ▶
Nodes	THF ▶	LSDA ▶	ASCII ▶
Solids	THF ▶	LSDA ▶	
Beams	THF ▶	LSDA ▶	
Shells	THF ▶	LSDA ▶	
Tk Shells	THF ▶	LSDA ▶	
Stonewalls	XTF ▶	LSDA ▶	ASCII ▶
Springs	XTF ▶	LSDA ▶	ASCII ▶
Seatbelts	XTF ▶	LSDA ▶	ASCII ▶
Retractors	XTF ▶	LSDA ▶	ASCII ▶
Sliprings	XTF ▶	LSDA ▶	ASCII ▶
Contacts	XTF ▶	LSDA ▶	ASCII ▶
Reactions	XTF ▶	LSDA ▶	ASCII ▶
Airbags	XTF ▶	LSDA ▶	ASCII ▶
Joints	LSDA ▶	ASCII ▶	
X Sections	LSDA ▶	ASCII ▶	
Subsystems	LSDA ▶	ASCII ▶	
Geo Contacts	LSDA ▶	ASCII ▶	
Nodal RBs	LSDA ▶	ASCII ▶	
Spotwelds	LSDA ▶	ASCII ▶	
SPCs	LSDA ▶	ASCII ▶	
BOUNDARYs	LSDA ▶	ASCII ▶	
FSIs	ASCII ▶		
SPHs	LSDA ▶	ASCII ▶	

Reset

## 5.17.2 Files

### File Names

By default the file filters in T/HIS are set to look for the file naming convention set for the LS-DYNA output files by the SHELL. This option can be used to swap the file filters back to the default LSTC naming convention. This option can be set in the [Preference File](#)

File	ARUP name	LSTC name
Time history	"jobname".thfd3thdt	
Extra Time history	"jobname".xtfxtfile	

### File Output

The [HIC](#), [3ms Clip](#), [ASI](#), [THIV](#), [TTI](#) Automotive injury criteria functions and [ERR](#) operator function can all send there output to a file as well as to the screen. These options can be used to select which functions send out to a file and to specify a Root Filename that is used for all of the output files. The Root Filename can be set in the [Preference File](#)

Settings

Data

Files

General

Layout

File Names

☒ ARUP (.thf, ...)
 ☐ LSTC (d3thdt, ...)

File Output

default

Root Filename for Output

☐ Write HIC to file
 

E:\Software Development\V

☐ Write 3ms Clip to fil
 

E:\Software Development\V

☐ Write ASI to file
 

E:\Software Development\V

☐ Write THIV to file
 

E:\Software Development\V

☐ Write TTI to file
 

E:\Software Development\V

☐ Write ERR to file
 

E:\Software Development\V

THF/d3thdt File

☐ Swap Title
 

5

"File Skip" number

Auto

File family size (MB)

XTF/xtfile File

☐ Swap Title
 

5

"File Skip" number

Auto

File family size (MB)

## 5.17.3 General

### Curve Operations

All of the [AUTOMOTIVE](#) filters are designed to filter curves using seconds as the time unit. This option can be used to automatically convert the x-axis values of any curves from milliseconds to seconds before applying one of the filters. If a curve is automatically converted then the output curve is also automatically converted back into milliseconds. This option can be set in the [Preference File](#)

All of the [AUTOMOTIVE](#) filters require curves with constant time intervals. This option can be used to specify a default time interval that will be used to automatically regularise a curve before it is filtered.

By default the [HIC](#) and [3ms Clip](#) functions calculate and report a value to the screen. These options can be used to make T/HIS display the peak values and the time windows they occur over. These options can be set in the [Preference File](#)

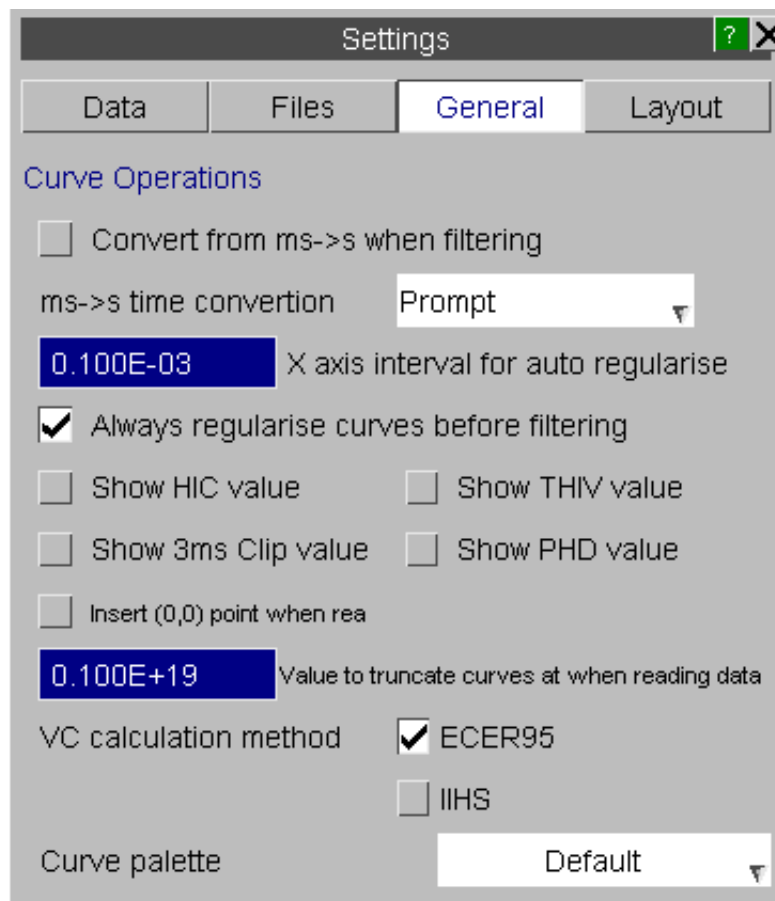
At present 2 different methods are used to calculate the VC injury criteria for the ECER95 and IIHS regulations (see [Appendix E](#) for more details ). This option can be used to set the default value. This option can be set in the [Preference File](#)

By default T/HIS uses 6 colours (White, Red, Green, Blue, Cyan and Magenta) for any curves that have not had a colour explicitly defined for them. Curves 1,7,13... will be White, 2,8,14... will be Red.

This option can be used to change the default number of colours T/HIS uses.

Default	Use the default 6 colours
Extended	Use the first 13 colours
No Grey	Use all 30 predefined colours except the 3 grey ones
Full	Use all 30 predefined colours plus any user defined ones.

The default value for the curve palette can also be specified in the "preferences" file (see [Appendix H](#) for more details).

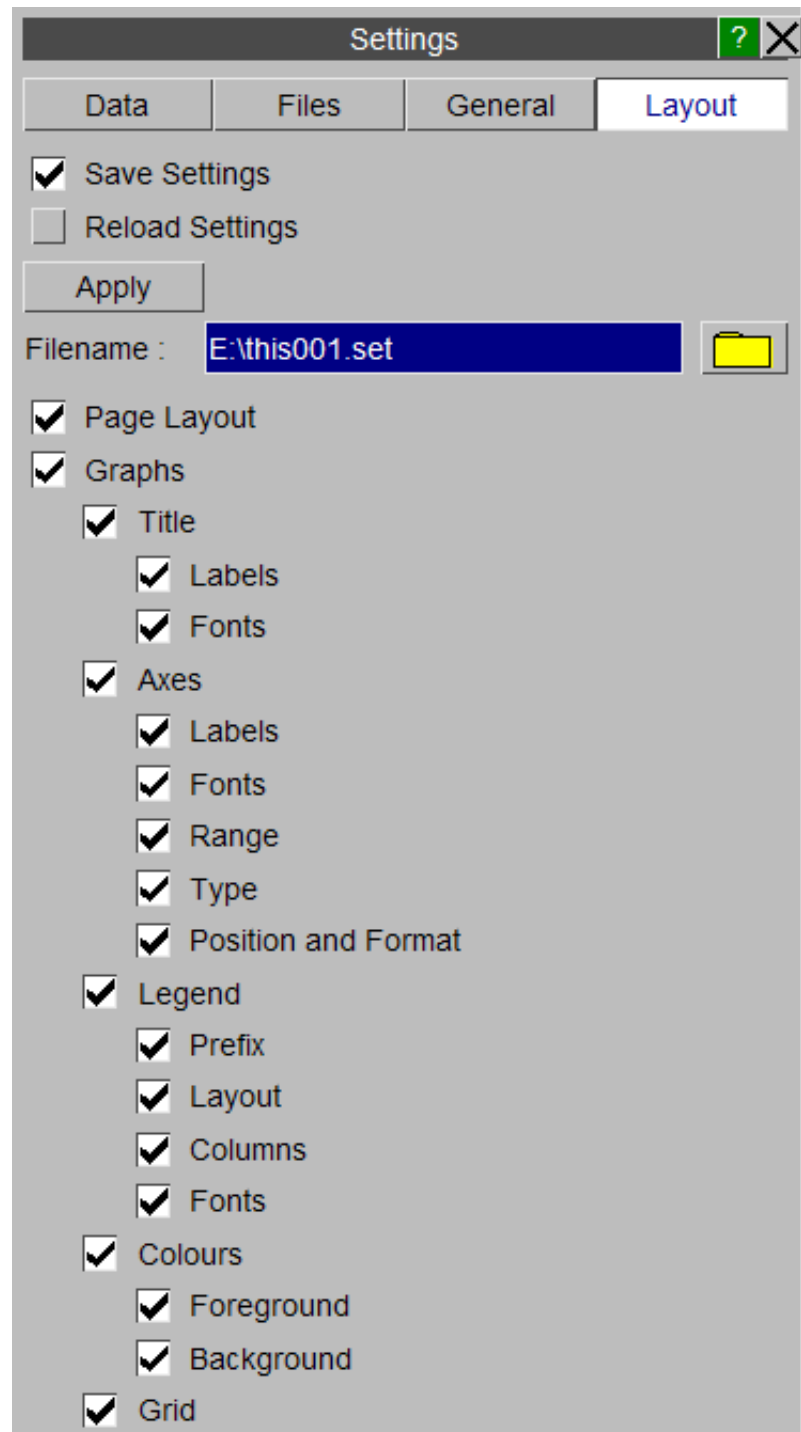


## 5.17.4 Layout

### Save Settings

This option can be used to save a T/HIS settings file which can be reloaded later. The settings file uses the same syntax as a FAST-TCF script except it only contains **layout** and **setup** commands.

The settings file can contain all of the commands required to reproduce the current page and graph layout or a subset of the commands.



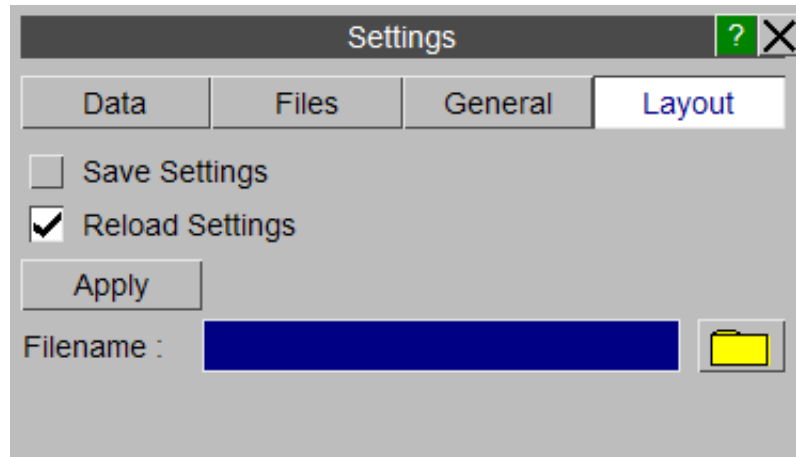
## Reload Settings

This option can be used to reload a previously saved settings file. In addition to reloading a file interactively a settings file can also be specified on the [command line](#)

-set=*filename*

or via the [Preference File](#)

this\*settings\_file: filename





## 5.18 MEASURE

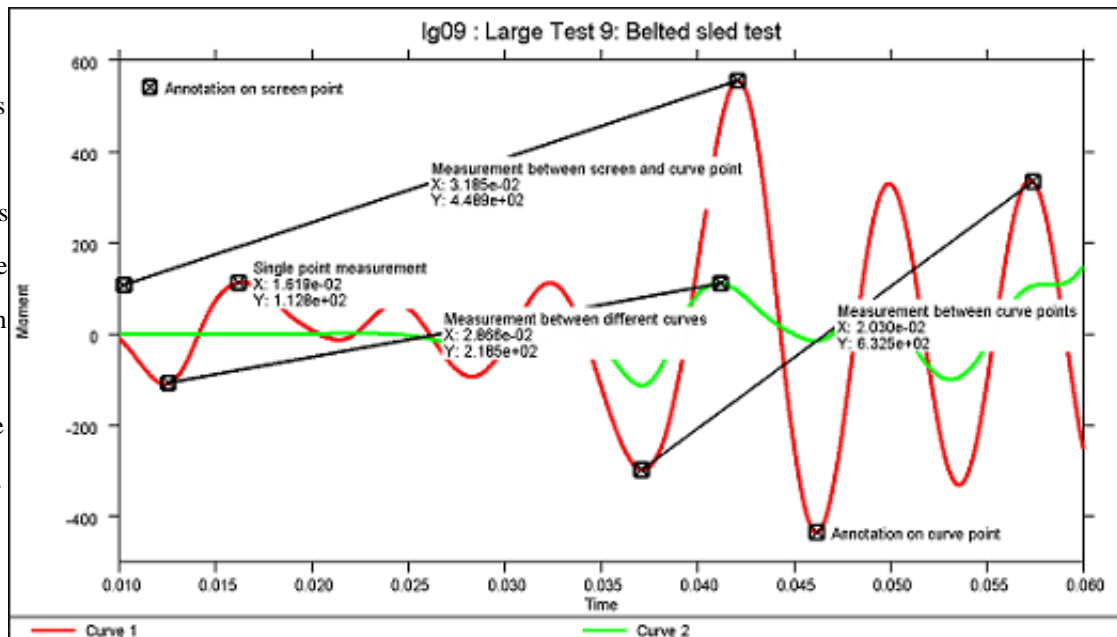
This menu can be used to make measurements between curve points and/or screen points. It can also be used to annotate graphs.

—	Read	Write	Curves	Models
Edit		Style	Properties	Images
Operate		Maths	Automotive	Seismic
Macros		FAST-TCF	Title/Axes	Display
Settings		Measure	Groups	Graphs
Command File		Units	JavaScript	Datum

Each graph  
can contain  
multiple  
measurements  
and  
annotations.

Measurements can be made between curve and/or screen points and can be made between different curves. Single points can be measured too.

Annotations  
can be made  
on curve or  
screen points.



### 5.18.1 Measure

Use this option to pick points on the graph to measure between.

Measure Menu

Measure Annotate

Navigation buttons: Back, Forward, First, Last

Delete All

P1: Curve Point P2: Curve Point

P1

P2

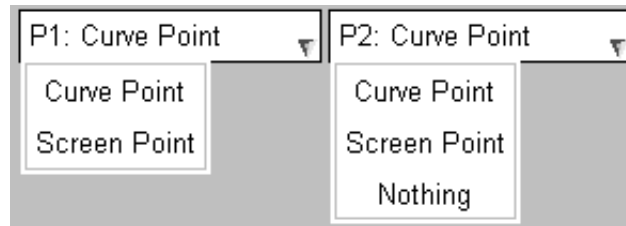
Distance

Label

Delete

## Point Types

Use the popups to select the point type to measure to/from.



## Label

Label

If you specify a label this will be displayed on the measurement.

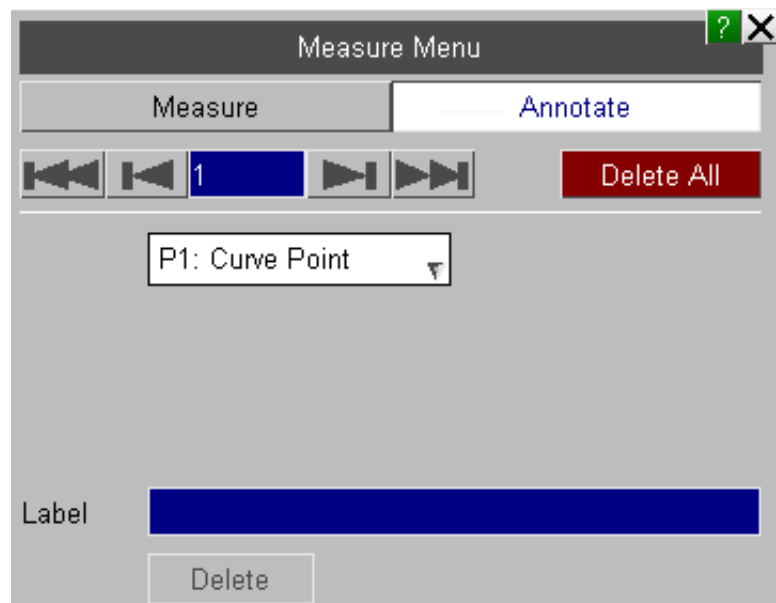
## Delete



This will delete the current measurement.

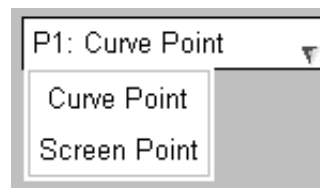
## 5.18.2 Annotate

Use this option to make annotations on the graph.



## Point Type

Use the popup to select the point type to annotate on.



## Label

Label

This is the annotation that will be displayed on the graph.

## Delete



This will delete the current annotation.

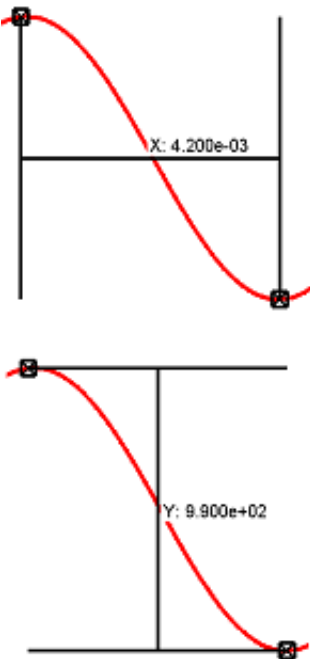
### 5.18.3 Format

These options can be used to control the display of the measurements and annotations.

#### Values

Measurements can be displayed with both the X and Y values, just the X value, just the Y value or neither.

If only one of the values is shown the line between the two points will be drawn like so:



<input checked="" type="checkbox"/> Show X Value	<input checked="" type="checkbox"/> Show Y Value
Current	Foreground
Other	
Text Font	Default
Text Size	automatic
Text Colour	Foreground
Background Colour	Background
	0 Transparency 100
	0
Border	<input type="checkbox"/> On/Off
Border Colour	Foreground
Border Width	
Symbol	<input checked="" type="checkbox"/> On/Off
Symbol Size	25
Number Format	Scientific (1.2345E)
Decimal Places	3

#### Text

The font, font size and colour of the values can be selected.

#### Background

To make it easier to read the values a background can also be specified. In addition to specifying the background colour a transparency value can be used to control the visibility of curves under the text.

#### Border and Border Colour

Specify a border and border colour to be added around the value.

#### Symbols

The symbols drawn on the measurement points can be turned on/off. The size of the symbol can also be specified.

## Number Format

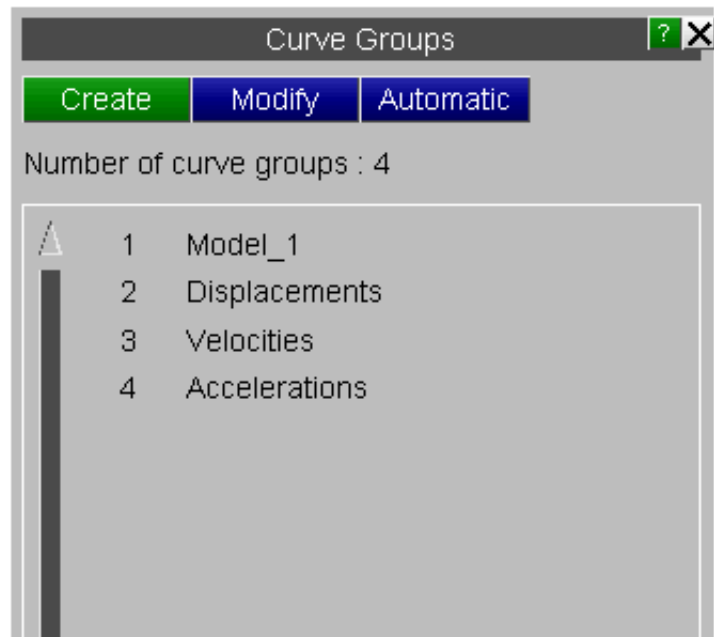
Specify the format of the values displayed on the graph.

## 5.19 Curve Groups

This panel can be used to create and modify curve groups. T/HIS can contain an unlimited number of curve groups each of which can contain any curve.

Curve groups can be used as input to most T/HIS functions that require one or more input curves ([see Section 5.0 for more details](#))

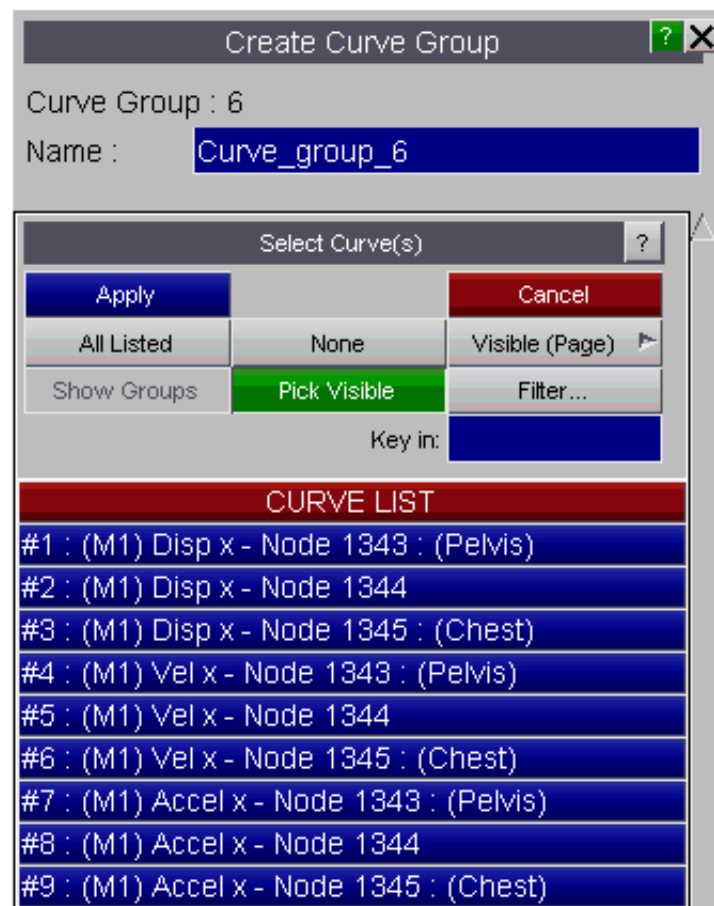
Each curve group should be given a unique name.



### 5.19.1 Create

This option can be used to create a new curve group.

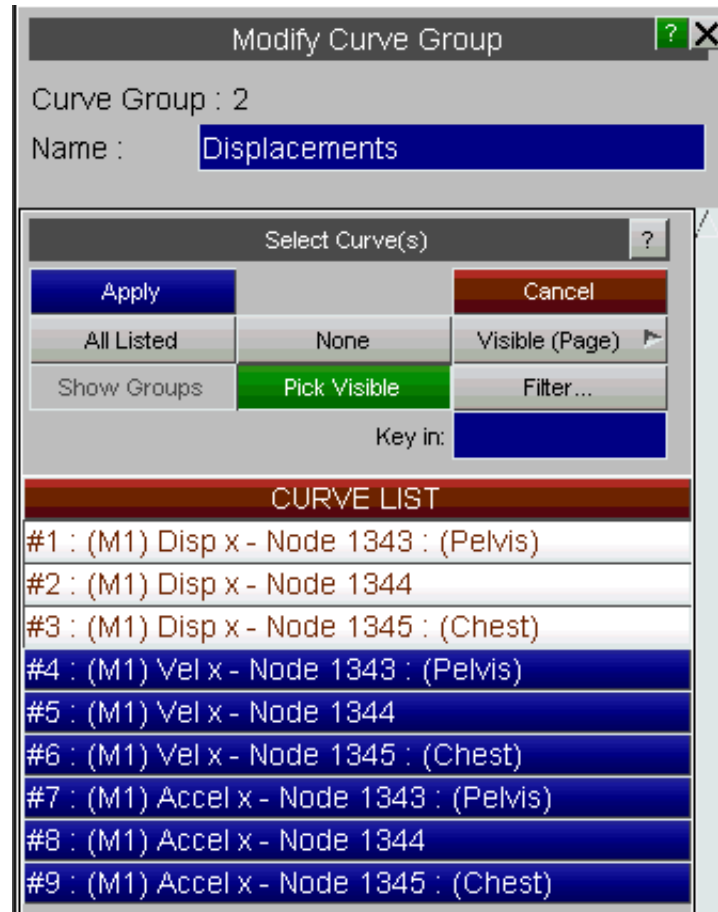
By default the group will be called "Curve\_group\_#" where "#" is the curve group number if an alternate name is not specified.



## 5.19.2 Modify

This option can be used to modify the contents of an existing curve group or its name.

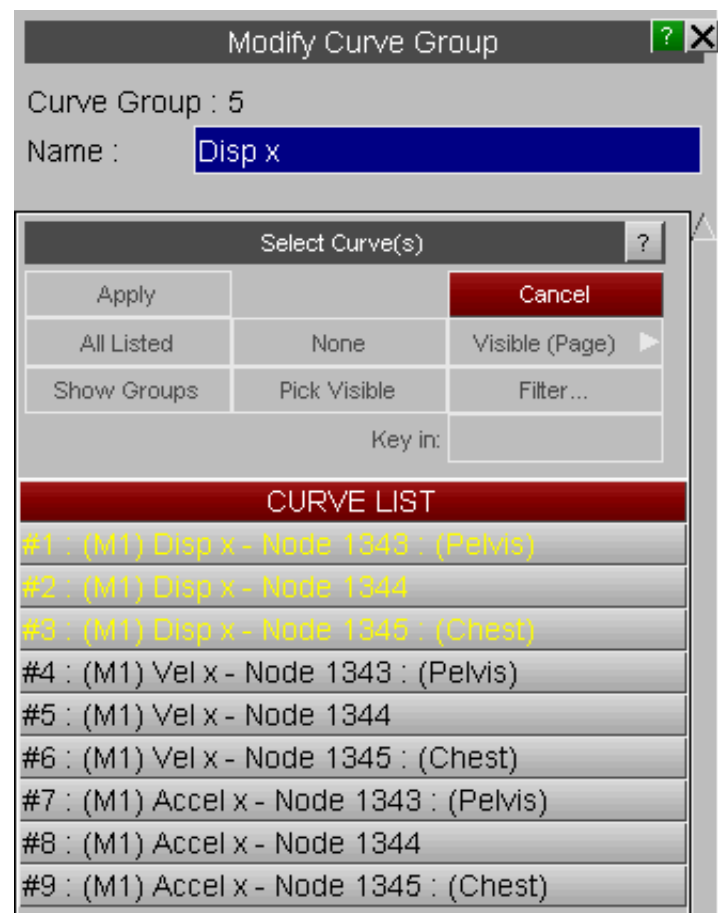
When a curve group is selected any curves that are already defined in the group are highlighted in the curve list.



The contents of [Automatic](#) curves groups can not be modified as T/HIS automatically adds and removes curves from automatic groups.

Curves that belong to an automatic curve group are highlighted in yellow.

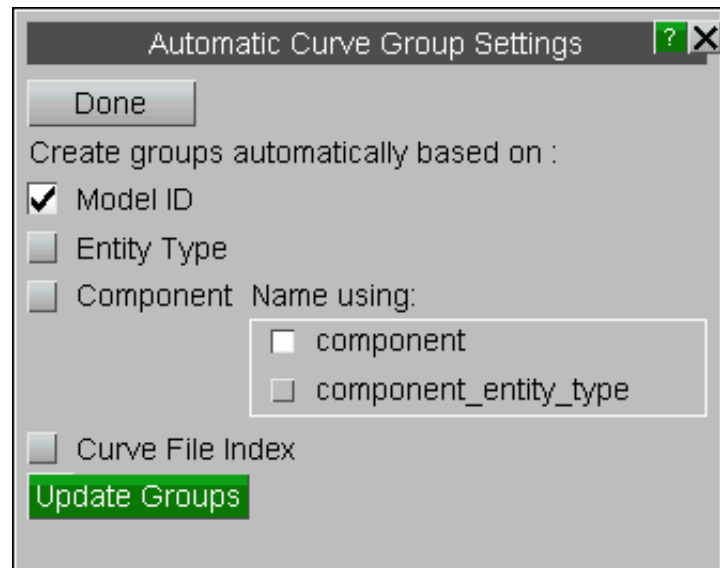
The name of an automatic curve group can be changed.



## 5.19.3 Automatic

By default T/HIS will automatically create a curve group for each model that is read in and will add any curves read in from that model into the curve group.

This option can be used to create other "automatic" curve groups.



Model ID	The default - one group is created for each model.
Entity Type	This option will create one group for each Entity type (Modal, Node, Solid etc) that data is read from. If data is read from multiple models then a single group for each entity type will be created containing curves from multiple models.
Component Name	This option will create one group for each component (Node X displacement, Contact X Force etc), that data is read from. If data is read from multiple models then a single group for each component will be created containing curves from multiple models.  The component groups can be named using either the component name (Disp X, Vel X...) or the component name and the entity type (Disp X - Node, Vel X - Node)
Curve File Index	If curves are read in from curve files (.CUR or CSV) then this option will create one group for the 1st curves read from each file, a second group for the 2nd curve read from each file and so on.

Multiple options can be selected at the same time.

**Update Groups** This option will create and update the contents of automatic curve groups if the options are changed.

The following preference options can be used to change the default options, (see [Appendix H](#) for more details).

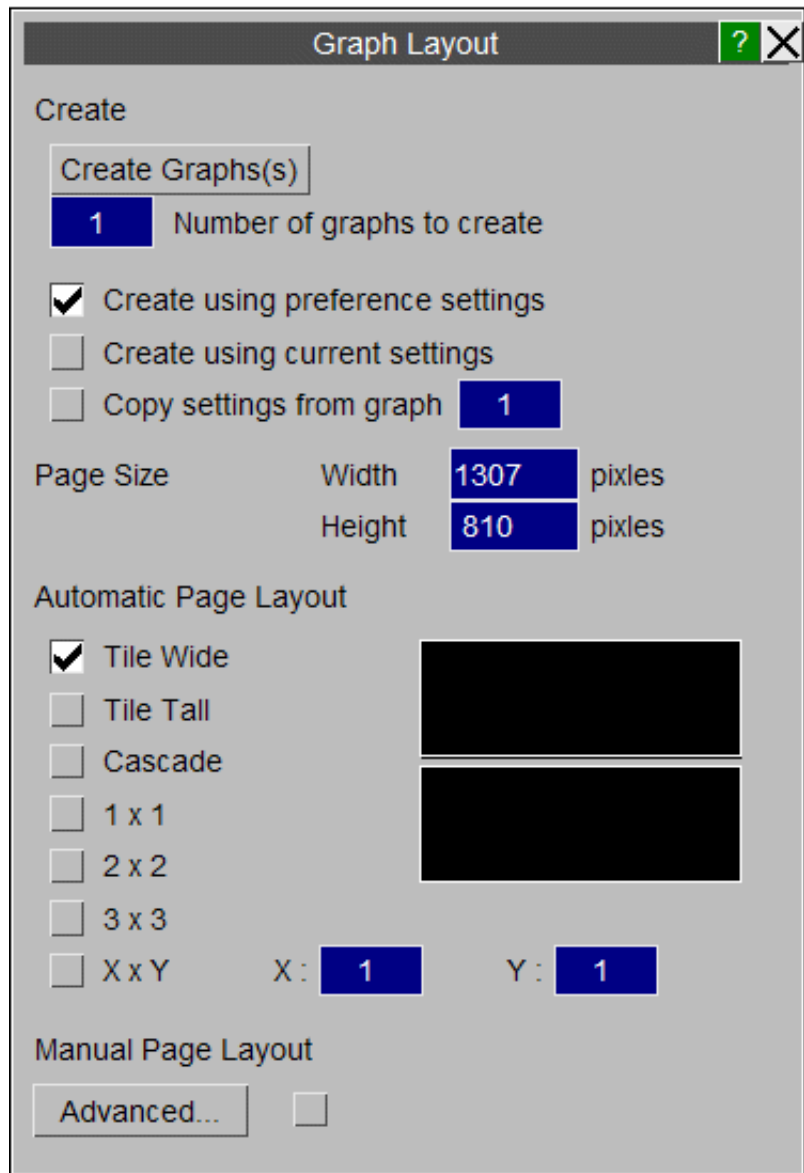
```
group_by_model
group_by_type
group_by_component
group_by_file_index
component_group_name
```

## 5.20 GRAPHS

This panel can be used to create additional graphs within T/HIS.

In addition to creating graphs this menu can also be used to control the layout of the graphs and to set up pages of graphs within T/HIS.

See [Section 3.0](#) for more details.



The image shows a 'Graph Layout' dialog box with a title bar containing a question mark icon and a close button. The dialog is organized into several sections. The 'Create' section at the top has a 'Create Graphs(s)' button, a numeric input for 'Number of graphs to create' set to 1, and three checkboxes: 'Create using preference settings' (checked), 'Create using current settings' (unchecked), and 'Copy settings from graph' (unchecked) with a graph index input set to 1. The 'Page Size' section contains 'Width' (1307 pixels) and 'Height' (810 pixels) inputs. The 'Automatic Page Layout' section includes checkboxes for 'Tile Wide' (checked), 'Tile Tall' (unchecked), 'Cascade' (unchecked), and grid layouts '1 x 1', '2 x 2', '3 x 3', and 'X x Y' (unchecked). To the right of these are two black rectangular preview windows. Below the grid layouts are 'X' and 'Y' position inputs, both set to 1. The 'Manual Page Layout' section at the bottom has an 'Advanced...' button and an unchecked checkbox.

**Graph Layout** ? X

Create

Create Graphs(s)

1 Number of graphs to create

☒ Create using preference settings

☐ Create using current settings

☐ Copy settings from graph 1

Page Size

Width 1307 pixels

Height 810 pixels

Automatic Page Layout

☒ Tile Wide

☐ Tile Tall

☐ Cascade

☐ 1 x 1

☐ 2 x 2

☐ 3 x 3

☐ X x Y X: 1 Y: 1

Manual Page Layout

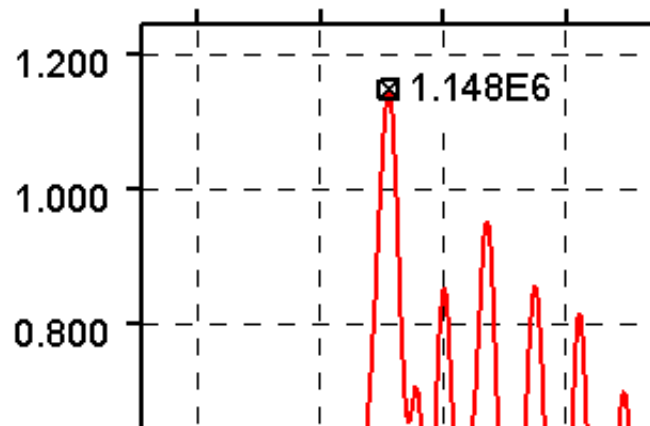
Advanced... ☐



## 5.21 PROPERTIES

This menu can be used to display addition curves properties.

Minimum and maximum curve values can be highlighted for each curve and the value can also be displayed.



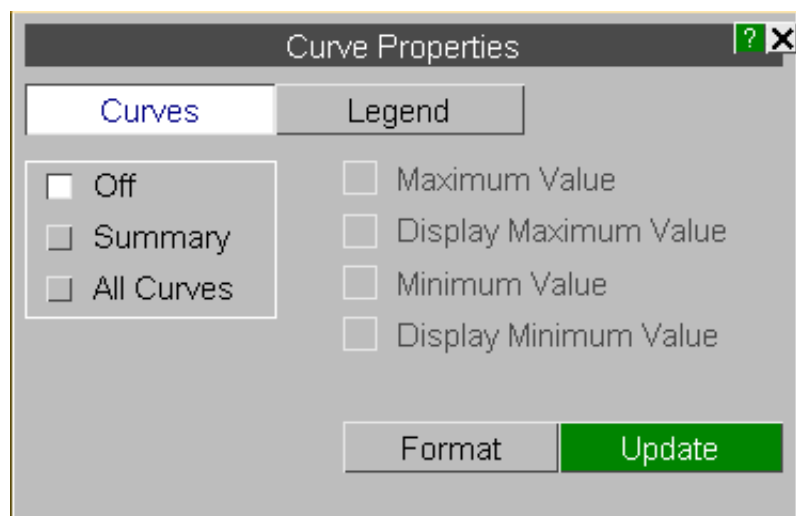
In addition to displaying the value on the curve the values can also be added automatically to the curve label in the graph legend.

— Node 1343 (max=1.148E6)

### 5.21.1 Curves

#### Curves (Off)

This option will turn off the display of all minimum and maximum values.



## Curves (Summary)

This option will display a single minimum/maximum value from all curves currently displayed..

The following properties can be displayed

<b>Maximum value</b>	Mark the maximum value with a cross
<b>Display Maximum</b>	Display the maximum value
<b>Minimum value</b>	Mark the minimum value with a cross
<b>Display Minimum</b>	Display the minimum value

Curve Properties

Curves Legend

☐ Off ☐ Maximum Value

☐ Summary ☐ Display Maximum Value

☐ All Curves ☐ Minimum Value

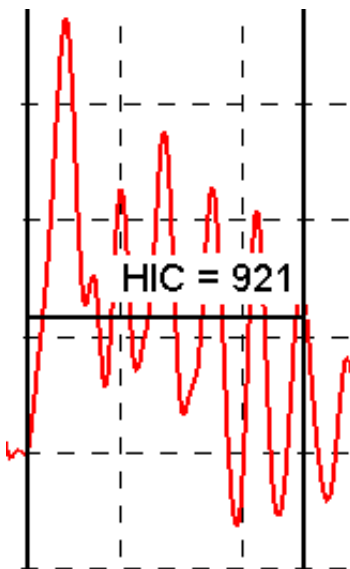
☐ Display Minimum Value

Format Update

## Curves (All curves)

This option can be used to select the properties that are displayed for each individual curve.

When this option is selected the display of injury criteria (HIC, HICd etc) for curves can also be selected.



Curve Properties

Curves Legend

☐ Off ☐ Summary ☐ All Curves

Format Update

Select All None

Pick Visible Show Curves

Highlight Max Show Max Value Highlight Min Show Min Value Show Hic Show 3ms Clip Show THIV Show PHD

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	#1 Accel mag - Node 1343 : (
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	#2 Accel mag - Node 1344
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	#3 Accel mag - Node 1345 : (
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	#4 Accel mag - Node 1346 : (
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	#5 Accel mag - Node 1347 : (

## 5.21.3 Format

This option can be used to control the display of the minimum/maximum values on the screen.

### Text

The font, font size and colour of the values can be selected. Either a single colour can be used for all the values or the values for each curve can be coloured using the same colour as the curve.

### Background

To make it easier to read the values a background can also be specified. In addition to specifying the background colour a transparency value can be used to control the visibility of curves under the text.

### Border and Border Colour

Specify a border and border colour to be added around the value.

### Connecting Line

This option will draw a line connecting the value with the point it relates to on the curve. The connecting line is drawn using the same colour as the border.

### Value

The values can be displayed showing just the Y axis value or with both the X and Y axis values. If both values are displayed they can either be displayed separated by a comma or one above the other.

The screenshot shows the 'Curve Properties' dialog box with the 'Curves' tab selected. The 'Done' button is highlighted in blue. The settings are as follows:

- Text Font:** Default
- Text Size:** Automatic
- Text Colour:** Foreground
- Background Colour:** Background
- Transparency:** A slider set to 0, with labels 0, Transparency, and 100.
- Border:** ☐ On/Off
- Border Colour:** Foreground
- Border Width:** A slider set to 1.
- Connecting Line:** ☐ On/Off
- Value:** ☒ Y Value Only, ☐ X, Y Values, ☐ X Value, ☐ Y Value
- Number Format:** Scientific (1.2345E+2)
- Decimal Places:** 3

## Number Format

The values can be displayed using 3 different formats

<b>Automatic</b>	Values are displayed using exponential format, all values are displayed as values of E0, E3, E6 etc. e.g 11.234E+03
<b>Scientific</b>	Values are displayed using exponential format. e.g 1.123E+04
<b>General</b>	Values are displayed as real numbers. e.g 11234.000

## Decimal Places

In addition to specifying the format, the number of decimal places can also be set between 0 and 9.

### 5.21.3 Legend

This option can be used to automatically added curve properties to the curve labels in the legend area.

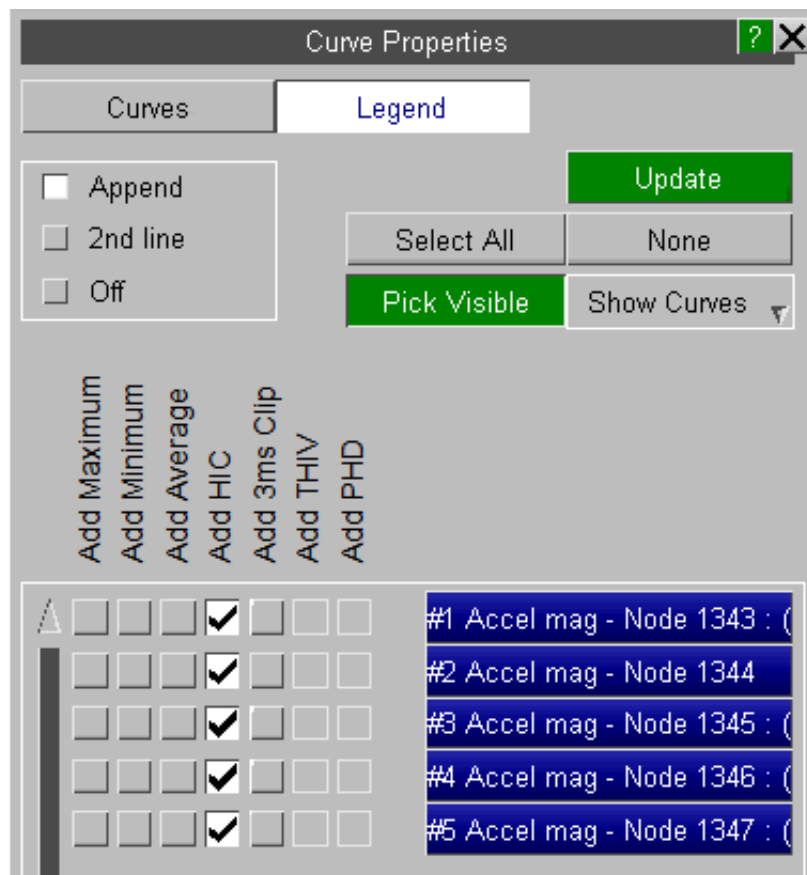
The following curve properties can be added to each curve label

Maximum value  
Minimum value  
Average value  
Injury Criteria (HIC, HICd etc)

Other options

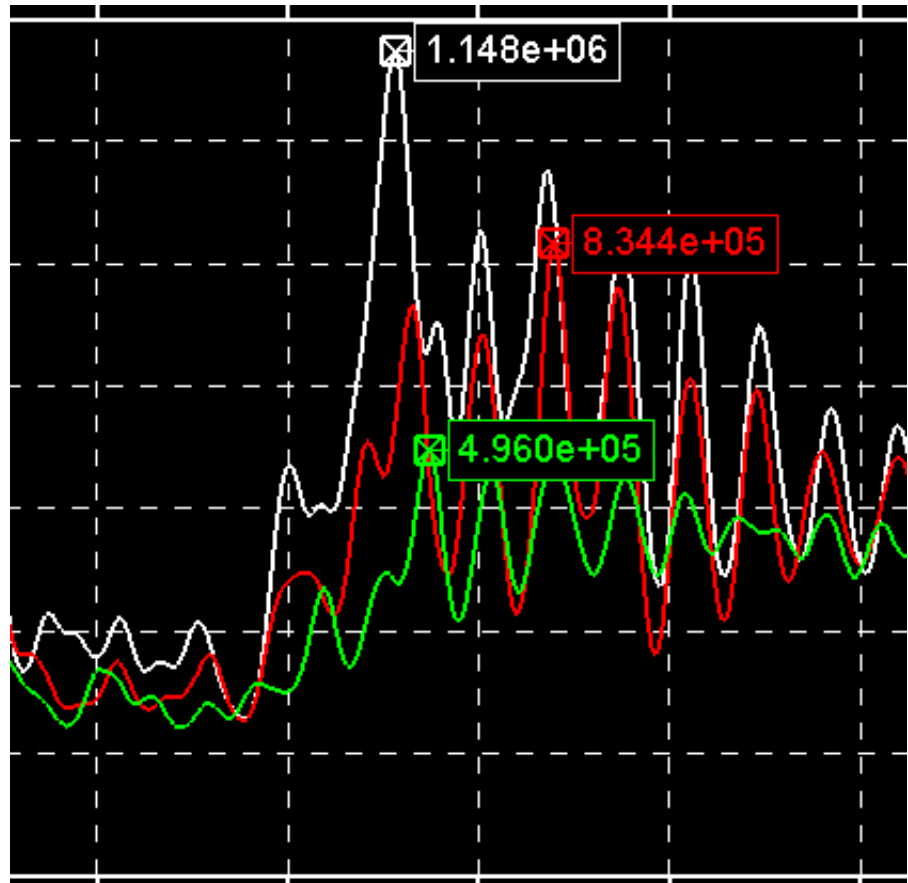
- Off** Turns off the display of curve properties in the legend
- Append** Add the values to the same line as the curve labels in the legend
- 2nd Line** Display the values using a second line for each curve in the legend

The format of the numbers added to the curve labels is the same as that used to display values on the curves.



## 5.21.4 Positioning Values

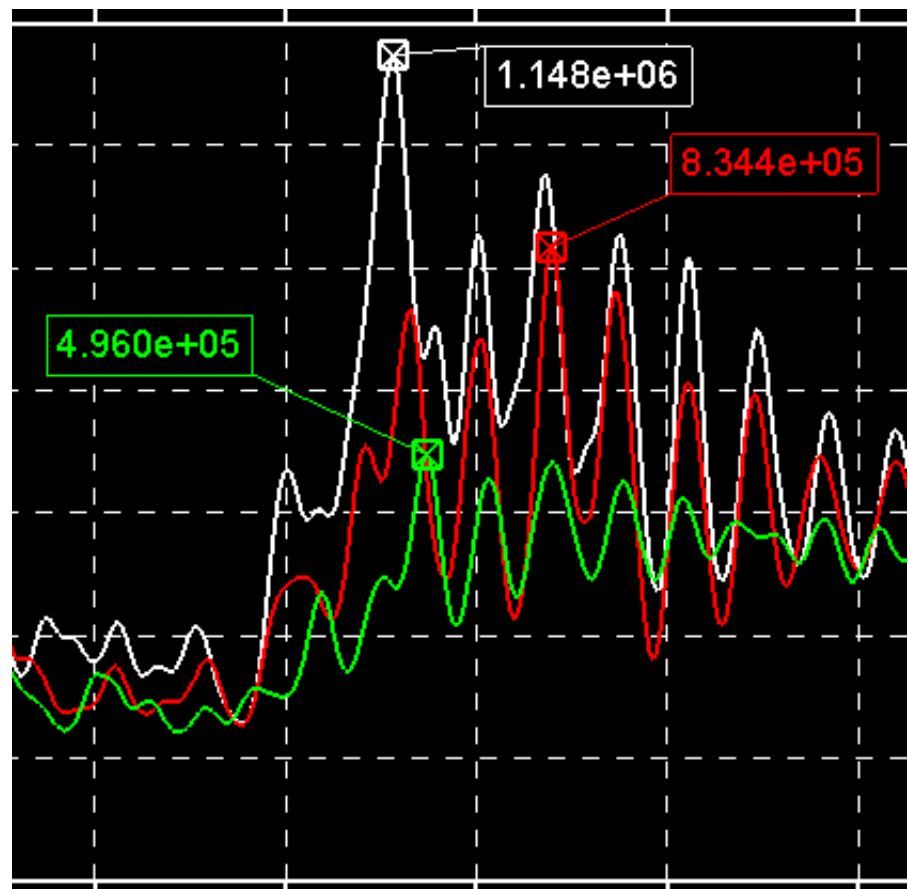
By default T/HIS will automatically position minimum and maximum values to the right of the point they apply to.



If the default location of the text obscures other curves then the position can be changed by clicking on the value with the left mouse button and then dragging the value to a new position.

If dynamic viewing is used to either zoom in or translate the curves after a value has been moved to a new position then it will maintain it's new position relative to the minimum/maximum value location.

As well as moving the minimum/maximum values the values used to display injury criteria like HIC and HIC(d) can also be moved.



## 5.22 UNITS

From version 9.4 onwards T/HIS tries to keep track of the units for each curves X and Y axis. For every data component that T/HIS can read from an LS-DYNA results file one of the following basic units is stored for the curves X and Y axis.

<b>Time</b>	<b>Rotation</b>	<b>Momentum</b>	<b>Energy Density</b>
<b>Energy</b>	<b>Rotational Velocity</b>	<b>Density</b>	<b>Mass Flow</b>
<b>Work</b>	<b>Rotational Acceleration</b>	<b>Stress</b>	<b>Frequency</b>
<b>Temperature</b>	<b>Length</b>	<b>Strain</b>	<b>Power</b>
<b>Displacement</b>	<b>Area</b>	<b>Force</b>	<b>Thermal Flux</b>
<b>Velocity</b>	<b>Volume</b>	<b>Moment</b>	<b>Force per unit width</b>
<b>Acceleration</b>	<b>Mass</b>	<b>Pressure</b>	<b>Moment per unit width</b>

When a curve operation is carried out on curve which has either the X or Y axis unit defined the units for the output curve(s) are also calculated. If a curve operation is carried out using 2 or more input curves with different units and the result is a curve with inconsistent units then the units are set to zero

If one of the inputs is a constant then it assumed to be unitless.

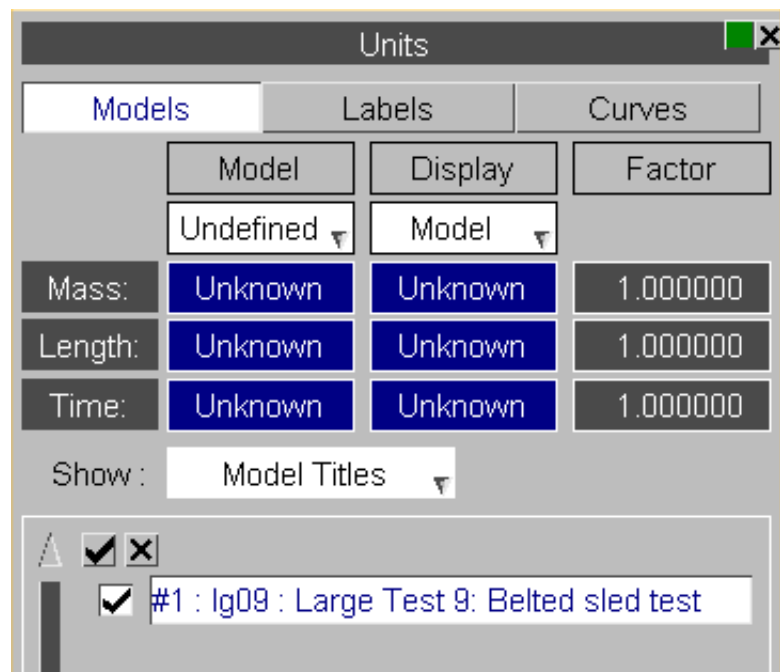
Input 1	Input 2	Operation	Output
Velocity (m/s)	Velocity (m/s)	Add	Velocity (m/s)
Velocity (m/s)	Displacement (m)	Add	Unknown
Velocity (m/s)	Velocity (m/s)	Divide	Constant
Velocity (m/s)	Displacement (m)	Divide	Frequency (1/s)
Velocity (m/s)	Constant	Add	Velocity (m/s)
Velocity (m/s)	Constant	Divide	Velocity (m/s)
Velocity (m/s)	-	Differentiate	Acceleration (m/s^2)

### 5.22.1 Models

By keeping track of the X and Y axis units for each curve T/HIS can now convert results from one unit system to another.

For each model one of the following 6 unit systems can be defined.

Name	Units
U1	m, kg, seconds (SI)
U2	mm, Tonnes, seconds
U3	mm, kg, milli-seconds
U4	mm, gm, milli-seconds
U5	ft, slug, seconds
U6	m, Tonnes, seconds



In addition to specifying a unit system for each model a separate unit system can also be selected to use to display results.

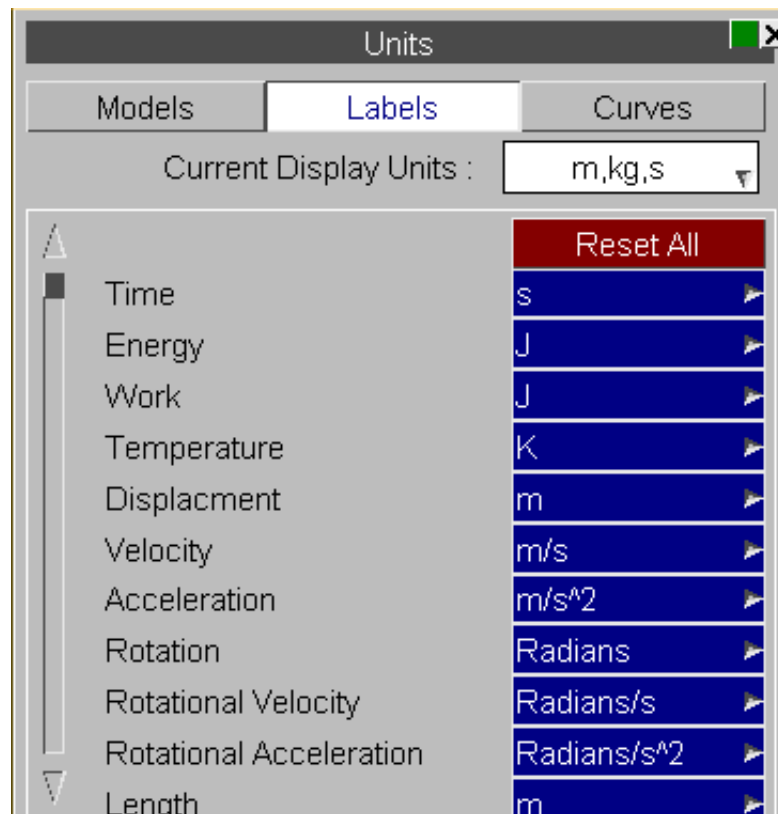
If the model unit system and the display unit system are different then T/HIS will automatically calculate the correct factors to apply to the X and Y axis as the curve data is read from the file (All curves are stored inside T/HIS using the currently defined Display unit system).

	Model	Display	Factor
	mm,T,s ▼	m,kg,s ▼	
Mass:	Tonne	Kilogram	1000.0000
Length:	mm	metre	0.001000
Time:	Sec	Sec	1.000000

## 5.22.2 Labels

This option will display the labels that will be used for each of the built in units. Each Unit System has it's own set of labels which can be modified if required.

The default labels for each unit system are shown below.



	U1: m,kg,s	U2: mm,T,s	U3: mm,kg,ms	U4: mm,gm,ms	U5: ft,slug,s	U6: m,T,s
<b>Time</b>	s	s	ms	ms	s	s
<b>Energy</b>	J	mJ	J	mJ	ft lbf	kJ
<b>Work</b>	J	mJ	J	mJ	ft lbf	kJ
<b>Temperature</b>	K	K	K	K	K	K
<b>Displacement</b>	m	mm	mm	mm	ft	m
<b>Velocity</b>	m/s	mm/s	mm/ms	mm/ms	ft/s	m/s
<b>Acceleration</b>	m/s <sup>2</sup>	mm/s <sup>2</sup>	mm/ms <sup>2</sup>	mm/ms <sup>2</sup>	ft/s <sup>2</sup>	m/s <sup>2</sup>
<b>Rotation</b>	Radians	Radians	Radians	Radians	Radians	Radians
<b>Rotational Velocity</b>	Radians/s	Radians/s	Radians/s	Radians/s	Radians/s	Radians/s
<b>Rotational Acceleration</b>	Radians/s <sup>2</sup>	Radians/s <sup>2</sup>	Radians/s <sup>2</sup>	Radians/s <sup>2</sup>	Radians/s <sup>2</sup>	Radians/s <sup>2</sup>
<b>Length</b>	m	mm	mm	mm	ft	m
<b>Area</b>	m <sup>2</sup>	mm <sup>2</sup>	mm <sup>2</sup>	mm <sup>2</sup>	sq ft	m <sup>2</sup>
<b>Volume</b>	m <sup>3</sup>	mm <sup>3</sup>	mm <sup>3</sup>	mm <sup>3</sup>	cu ft	m <sup>3</sup>
<b>Mass</b>	kg	T	kg	gm	slug	T
<b>Momentum</b>	kg m/s	T mm/s	kg mm/ms	gm mm/ms	ft slug/s	T m/s
<b>Density</b>	kg/m <sup>3</sup>	T/mm <sup>3</sup>	kg/mm <sup>3</sup>	gm/mm <sup>3</sup>	slug/cu ft	T/m <sup>3</sup>
<b>Stress</b>	N/m <sup>2</sup>	N/mm <sup>2</sup>	kN/mm <sup>2</sup>	N/mm <sup>2</sup>	lbf/sq ft	kN/m <sup>2</sup>
<b>Strain</b>	-	-	-	-	-	-
<b>Force</b>	N	N	kN	N	lbf	kN
<b>Moment</b>	Nm	Nmm	kNmm	Nmm	ft lbf	kNm
<b>Pressure</b>	N/m <sup>2</sup>	N/mm <sup>2</sup>	kN/mm <sup>2</sup>	N/mm <sup>2</sup>	lbf/sq ft	kN/m <sup>2</sup>
<b>Energy Density</b>	J/m <sup>3</sup>	mJ/mm <sup>3</sup>	J/mm <sup>3</sup>	mJ/mm <sup>3</sup>	ft lbf/cu ft	kJ/mm <sup>3</sup>
<b>Mass FLOW</b>	kg/s	T/s	kg/ms	gm/ms	slug/s	T/s
<b>Frequency</b>	Hz	Hz	kHz	kHz	Hz	Hz
<b>Power</b>	W	mW	kW	W	ft lbf/s	kW
<b>Thermal Flux</b>	W/m <sup>2</sup>	mW/mm <sup>2</sup>	kW/mm <sup>2</sup>	W/mm <sup>2</sup>	lbf/ft	kW/m <sup>2</sup>
<b>Force per unit width</b>	N/m	N/mm	kN/mm	N/mm	lbf/ft	kN/m
<b>Moment per unit width</b>	Nm/m	Nmm/mm	kNmm/mm	Nmm/mm	ft lbf/ft	kNm/m

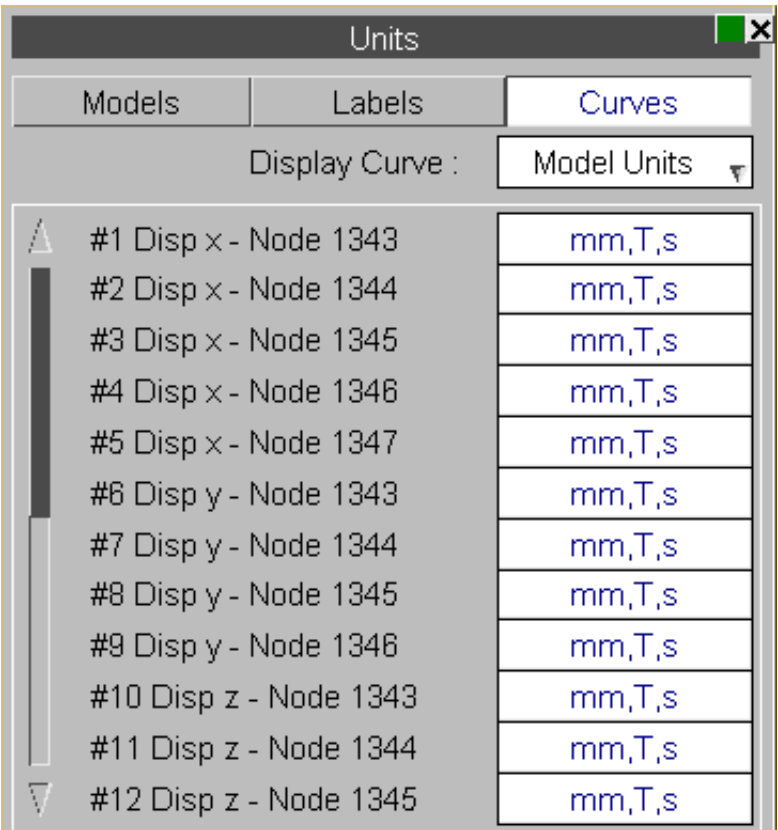
If a curve has a user defined unit or if after a curve operation one of the curve axis unit is not one of the basic units that T/HIS knows about then T/HIS will build a label from the currently defined length,mass,time,temperature and angle labels.

If for example a velocity/time curve is multiplied by another velocity time curve then the Y axis will have units of Velocity<sup>2</sup>. If the current display unit system is U1 (m,kg,seconds) then the unit label for the curves y axis will be "m<sup>2</sup>/s<sup>2</sup>".

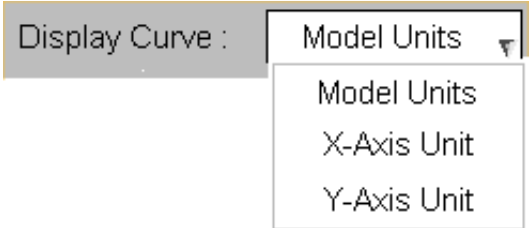


### 5.22.3 Curves

This option can be used to display the unit information for each curve.



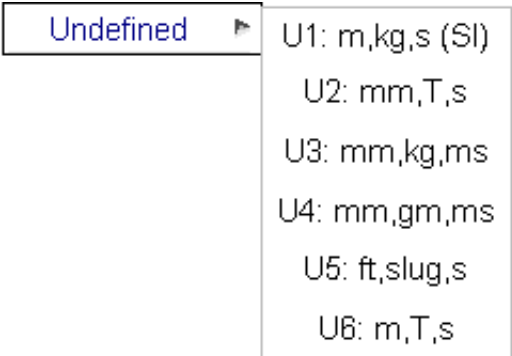
By default the unit system for each curve is displayed but his can be changed to show either the X or Y axis unit using the popup menu.



### Setting the Unit System for a Curve

If the unit system for a curve has not been defined then it will be displayed as "Undefined" and a popup menu will be available that can be used to select the correct unit system. If the selected unit system is different to the unit system currently being used to display results then the curve values will automatically be converted to the current display unit system.

**Note :** *Once the unit system for a curve has been defined it can not be changed.*



## Setting the Axis Units for a Curve

The X and Y axis units of a curve can be defined or changed at any time. The popup menu contains all of the basic Unit types that T/HIS knows about plus an option to setup a user defined unit.

To create a user defined unit for a curve the unit should be defined in terms of it's basic properties. The values for **mass, length, time, angle** and **temperature** should be the powers that are used to describe the unit in terms of it's fundamental dimensions.

Some examples of common units defined using this method are shown below.

Unit	Mass	Length	Time	Angle	Temperature
Time	0.0	0.0	1.0	0.0	0.0
Displacement	0.0	1.0	0.0	0.0	0.0
Velocity	0.0	1.0	-1.0	0.0	0.0
Acceleration	0.0	1.0	-2.0	0.0	0.0
Stress	1.0	-1.0	-2.0	0.0	0.0

Moment per Unit Width

Force per Unit Width

User Defined

Unit Label :	s
Length :	0.00
Mass :	0.00
Time :	1.00
Angle :	0.00
Temp :	0.00
Apply	

## 5.23 The Javascript Interface

### 5.23.1 Introduction

Javascript is a freely available scripting language that is normally found performing the "work" behind interactive web pages, however its syntax and structure also make it an excellent tool for providing an externally programmable interface to programmes in general.

Within T/HIS it is implemented as an Application Programming Interface (API) which provides a range of functions that allow you to edit and create curves, open windows, generate plots, and so on. This is written in a very simple and non-intimidating way, with relatively few functions, that should be easy for non-programmers to use.

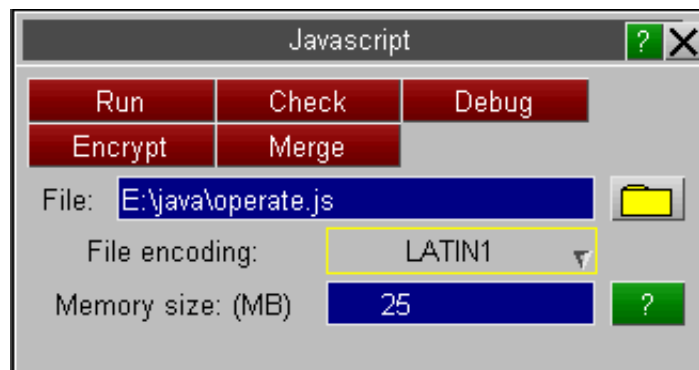
Anyone familiar with C or shell script programming will find existing Javascripts are instantly readable, and can be given minor edits without further ado. For those who are more ambitious a good guide to the language is "**Javascript, A definitive Guide**" by David Flanagan, published by O'Reilly, ISBN 0596101996.

The sections below describe how to run Javascripts in T/HIS, and summarise its Javascript API. For details of the API and its functions, and also some examples, see the [Appendix J](#)

### 5.23.2 Using Javascript in T/HIS.

Human-readable Javascripts need to be **compiled**, meaning turned from something human-readable into a set of instructions that a computer can understand; and then **run** in their compiled form. They can be changed and rerun in their modified form at any time without having to exit and re-enter T/HIS, making the "write, test, modify, re-test" development cycle very quick and easy.

#### 5.23.2.1 Compiling and Running a script



**Run** Will both compile and run the script unless it contains syntax errors, in which case it stops with an error message when compilation fails.

**Check** Only compiles the script, reporting any errors found, and does not run it.

**Encrypt** A script can be encrypted so that the source code is hidden but the script can still be run (when compiling and running the script T/HIS decrypts the file in memory). Once encrypted the source code cannot be retrieved by an ordinary user so make sure that you keep the original file somewhere safe. As a last resort contact OASYS Ltd who can decrypt the script if required.

If a script is split up into separate files by Use the files are all combined together into the main file before encrypting.

**Merge** If a script is split up into separate files by Use the files are all combined together into a single file. This may be useful if you want to give the script to someone else and you do not want to have to give lots of different files.

**Debug** Starts the JavaScript debugger, [JaDe](#) to debug the script.

**Memory size** is the memory allocated for garbage collection in the JavaScript engine. Please see the [garbage collection section](#) for more details.

### File encodings for scripts

Version 10.0 of T/HIS introduced the ability for unicode text to be used on widgets created in a script. Previous versions of T/HIS only supported English text so the default ASCII encoding was used for script files (this is still the

default encoding for script files).

If you want to use unicode text in widgets then you must use a file encoding that is capable to representing the unicode 'characters' you require. The **File encoding** popup allows you to change the file encoding used when reading the script file. T/HIS supports the following file encodings:

Encoding	Description
LATIN-1	Default 'ASCII' encoding
BIG5	Taiwan/Hong Kong (traditional)
EUC-CN	Extended unix code (Simplified Chinese)
EUC-JP	Extended unix code (Japanese)
EUC-KR	Extended unix code (Korean)
GB	Chinese (simplified)
GBK	Chinese
ISO-2022-CN	Chinese
ISO-2022-CN-EXT	Chinese (extended)
ISO-2022-JP	Japanese
ISO-2022-JP-2	Japanese (extended)
ISO-2022-KR	Korean
JOHAB	Korean
SHIFT-JIS	Japanese
UTF-8	Should NOT have a byte order mark (BOM).
UTF-16	Should have a byte order mark (BOM). If not present assumes big endian
UTF-16LE	Little endian with or without byte order mark (BOM)
UTF-16BE	Big endian with or without byte order mark (BOM)
UTF-32	Should have a byte order mark (BOM). If not present assumes big endian
UTF-32LE	Little endian with or without byte order mark (BOM)
UTF-32BE	Big endian with or without byte order mark (BOM)

Please contact Oasys Ltd if you have problems or require another encoding to be supported.

To show the unicode text the appropriate font must be used. This can be set using the preferences `primer*cjk_unix_font` and `primer*cjk_windows_font`.

### 5.23.2.2 Dealing with errors in scripts

Script errors come in two forms:

#### **Syntax errors**

Are mistakes of Javascript grammar or spelling, resulting in error messages during compilation.

These are easy to detect and correct since the line number and offending syntax are both described by the compiler. The script needs to be edited to correct the problem and then recompiled. Sometimes several iterations of the compile/edit cycle are required to eliminate all errors from a script.

**Run-time errors**

Are errors of context or logic in scripts that are syntactically correct, and thus have compiled, but which fail at some stage when being run.

A typical example of a run-time error is an attempt to divide a value by zero, yielding the illegal result infinity. More subtle errors involve passing an invalid value to a function, accessing an array subscript that is out of range, and so on.

The T/HIS Javascript API has been written in such a way that it handles "harmless" run-time errors by issuing a warning and continuing execution, but that more serious errors which could result in the wrong answers being generated issue an error message and terminate.

### 5.23.2.3 Setting the Garbage Collection Threshold Size

Memory size: (MB)

25

?

(This is an advanced topic, and you don't need to understand it.)

JavaScripts execute inside a memory "arena", allocated dynamically from the operating system, which grows in size as storage is requested within the script. This growth occurs due to requests for "new" variables within the script and also when API functions allocate and return values and objects, and it is limited only by what the operating system can deliver.

The nature of JavaScript means that objects frequently become redundant, and it is wasteful not to reuse the storage that they occupy, therefore there is a "Garbage Collection" process running behind the scenes which periodically checks storage and releases that which is no longer needed. This process is automatic and hidden from the user, it just "happens".

However Garbage Collection is quite a CPU-hungry process, so it is only carried out periodically when a certain threshold is reached. This can sometimes be observed during script execution as a periodic "pause for thought", and if you are monitoring memory usage with a system tool you may see it drop during these pauses.

Clearly this threshold value must be large enough not to trigger excessively frequent (and costly) garbage collections, while at the same time not being so large that scripts build up large amounts of excess memory to the detriment of the rest of the programme.

The **Memory size** value in the JavaScript panel is the amount of memory allocated for garbage collection. Every time a new object, array, string or double precision number is used a garbage collection 'thing' is also allocated. The Memory size is the total memory for these 'garbage collection things', **NOT** the total memory for the script. The total memory for the script could be significantly higher than this value. e.g the memory required for a Model object could be several kbytes but the memory for the 'garbage collection thing' for the Model object will something like 10 bytes for a 64bit operating system.

When the memory used for garbage collection 'things' reaches a significant proportion of **Memory Size** (normally about 2/3) then garbage collection will take place to try to reclaim memory. If no memory can be reclaimed and the total memory used for garbage collection reaches **Memory size** then the script will terminate with an error.

If your script has to retain a large number of objects, arrays, strings etc in memory then you may have to increase the value for **Memory size**. This can also be done using the `this*javascript_memory_size` preference or adding a special [memory comment](#) at the top of the script.

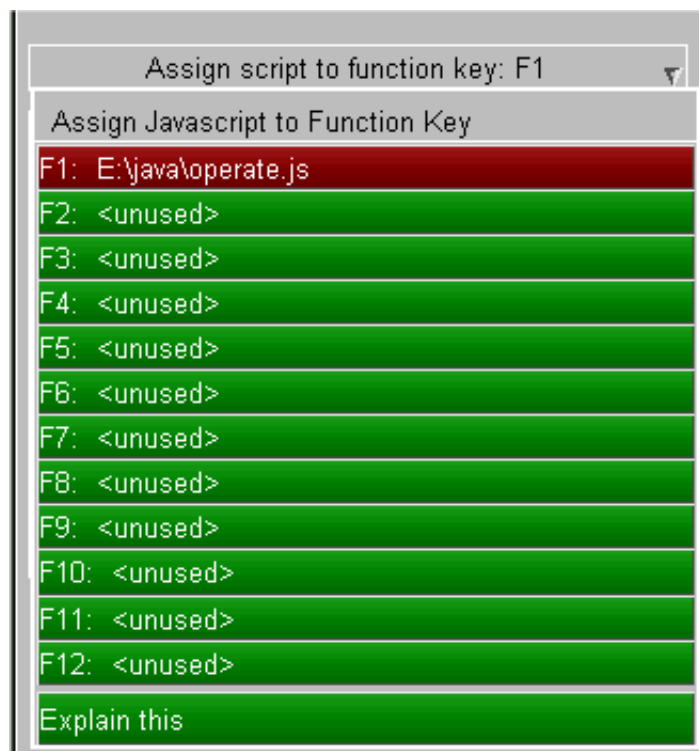
To recap:

- This threshold does *not* limit the memory the script can use, that is limited only by the operating system.
- It sets the memory for Garbage Collection 'objects'.
- Scripts which allocate a lot of memory, and which exhibit frequent pauses, *may* run faster with a larger value.
- ... and finally:

If you don't understand this topic don't worry. Most scripts will run quite happily with the default value, and you can ignore this setting unless they appear to be struggling, in which case try raising it. (As good an approach as any is to keep on doubling this value until the script works, but don't use very large sizes unnecessarily.)

### 5.23.2.4 Assigning Javascripts to Shortcut Keys

If a script is to be run repeatedly it can be convenient to set up a short-cut to it. From within the JavaScript menu this script can be assigned to one of the 12 function keys, alternatively the JavaScript can be assigned to any key using the [Shortcut menu](#).



### 5.23.2.5 Maintaining a library of Javascripts

It is also convenient to have a library of scripts in a defined location.

By default T/HIS looks in `$OA_INSTALL/this_library/scripts`, but you can define a different directory by setting the preference:

```
this*script_directory:
some_different_directory_name
```

In the your `oa_pref` file.

All scripts found in the relevant directory will be listed in the Javascript panel, as shown in this example.



#### Using the "description:" comment at the top of a script to identify its purpose.

To help to identify scripts special comments are searched for in the top 10 lines of each script, and if `description:` is found, for example the comment line:

```
// description: Some description of the script's purpose
```

Then the description line is shown as hover text when the mouse is placed over that filename. In the example above the "princ2d" script has the line

```
// description: Colour curve by model number
```

#### Using the "name:" comment at the top of a script to change its name

Normally the name shown for a script will be its filename, stripped of any leading pathname and trailing ".js" extension.

However if the string **name:** is found in the first ten lines of the script, then the following name will be used instead. For example the line:

```
// name: Colour By Model
```

Will result in the script appearing with the name "Colour By Model" in the Javascript panel. This does not affect the actual name of the script, only the name on its library button.

### Using the "memory:" comment at the top of a script to change the required memory

Sometimes the [memory required for garbage collection](#) needs to be changed.

If the string **memory:** is found in the first ten lines of the script, then the size given will be used for the memory (unless the size in the memory textbox is larger than this value). For example the line:

```
// memory: 50
```

Will result in the script using 50Mb for garbage collection memory.

## 5.23.3 Running a Javascript in "batch" mode.

All the above assumes that Javascripts will be run interactively from the user interface, however it is also possible to run a script in "batch" mode using the command line interface. The relevant command-line commands are:

```
/JAVASCRIPT - +- COMPILE      Compiles and checks the script, but does not run it.
               +- EXECUTE      (Re)compiles and runs the script
               +- MEMORY <nnn> Resets the Garbage Collection threshold to <nnn> MBytes
```

To run a Javascript from batch these commands need to be placed in a command file and run using the command line "-cf=command filename" option. For example the command file might be:

```
... some other commands
/JAVA EXEC my_script.js
...some further commands
```

And the command line required to run T/HIS might be something like:

```
$OASYS/this10.exe -d=default -cf=command_file -exit analysis_name
```

Obviously multiple script invocations may be placed in a command file. For more information see:

[Command and Session files](#) Describes command files, and explains how to create and use them

[T/HIS command line arguments](#) Describes the various command line arguments, and how to use them

## 5.23.4 Running a Javascript from with a FAST-TCF script

JavaScripts can also be run from within a FAST-TCF script using the "javascript" option

```
javascript "E:\javascripts\new_function.js"
```

Within a FAST-TCF script curves are usually accessed via curve tags. If a JavaScript is used within a FAST-TCF script it is recommended that the **Curve.GetFromTag()** function is used to access existing curves. If a new curve is created by a JavaScript within a FAST-TCF script then the new curve can be accessed within the FAST-TCF script using the "tag" parameter of the curve creation function

```
new_curve = new Curve(id,tag,label,x-axis label,y-axis label);
```

If a tag is not specified in the curve creation function

```
new_curve = new Curve(id);
```

then a curve tag will be generated automatically for the curve. The 1st curve created within the script will be tagged "curve\_js\_1", the 2nd "curve\_js\_2" ...

## 5.23.5 The T/HIS Javascript API

All of the T/HIS JavaScript API functions are described in detail in Appendix J.

## 5.23.6 Examples

By far the easiest way to learn Javascript is by example and, more specifically by modified existing scripts to do what you want.

The software comes supplied with examples in the **\$OASYS/programme\_library/examples** directory (for T/HIS **\$OASYS/this\_library/examples**) and you are free to use and modify these files for your own purposes.

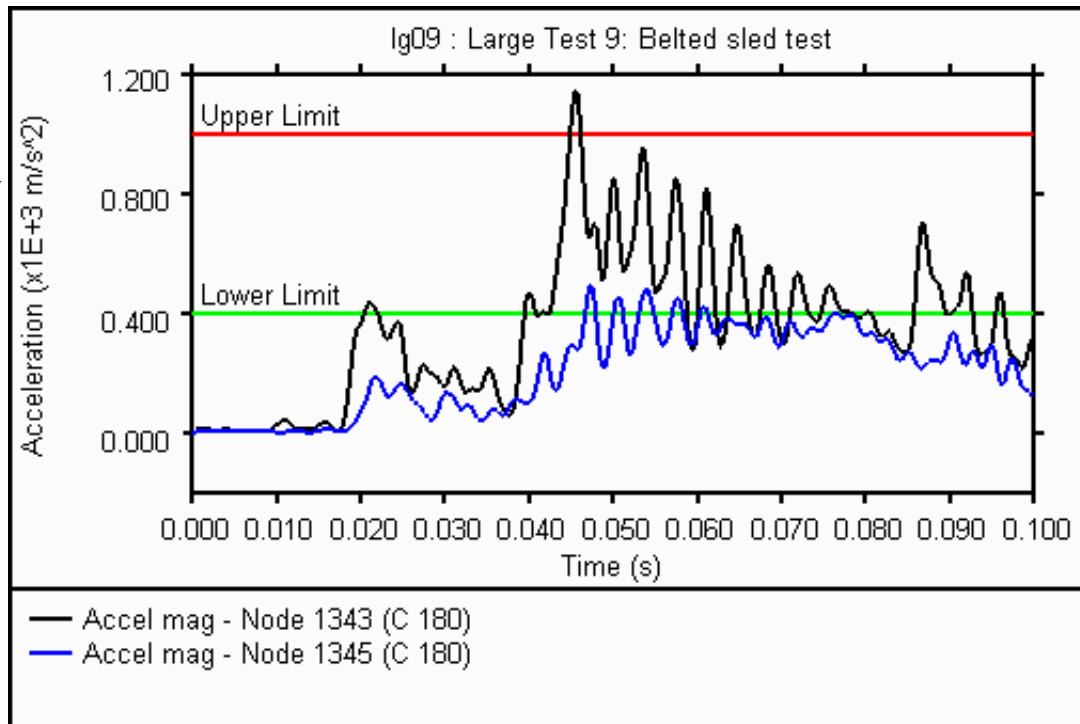


## 5.24 Datum Lines

Datum lines can be added to graphs to show limits and reference curves. Unlike normal curves DATUM lines are not used to calculate graph limits when auto scaling and are not shown in the curve legend.

—	Read	Write	Curves	Models
Edit	Style	Properties	Images	
Operate	Maths	Automotive	Seismic	
Macros	FAST-TCF	Title/Axes	Display	
Settings	Preferences	Groups	Graphs	
Command File	Units	JavaScript	Datum	

Each graph can contain multiple DATUM lines, all DATUM lines are drawn in the order they have been defined before any curves are plotted.

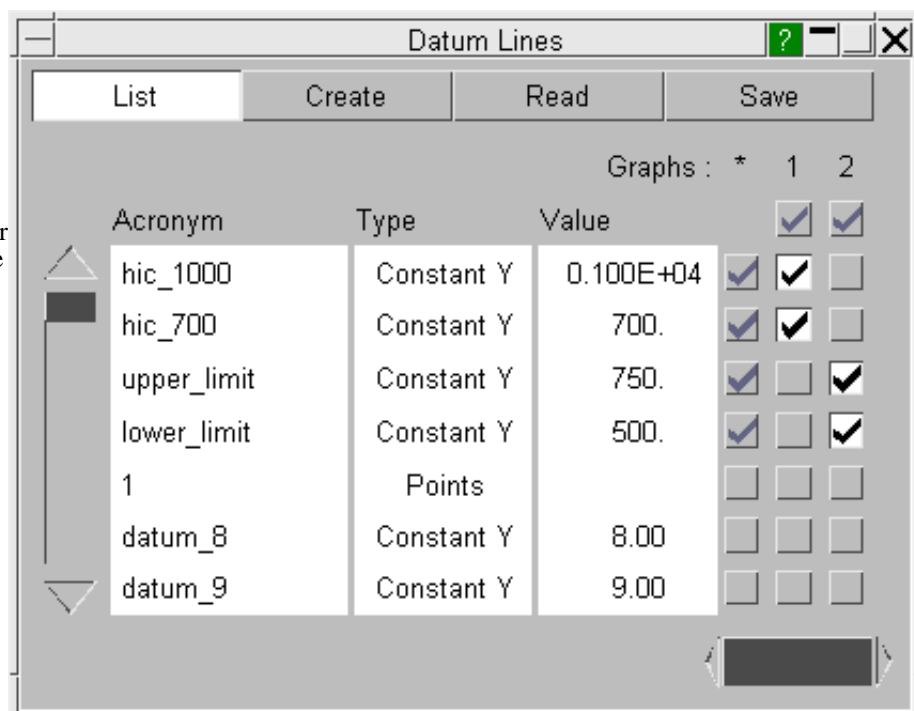


### 5.24.1 List

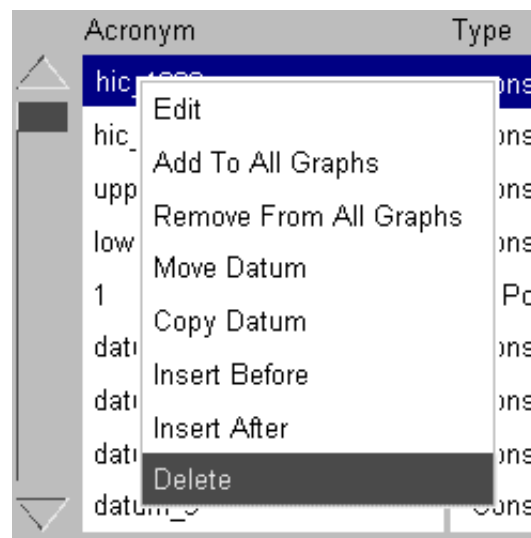
This option lists all DATUM line definitions that have been created.

This menu can also be used to select which DATUM lines appear on each graph. Each DATUM line can appear on more than one graph.

A range of DATUM lines can be added/removed from graphs by selecting the first line/graph combination and then holding down SHIFT while selecting the second line/graph.



Clicking on any of the DATUM line definitions will highlight it in blue and display a popup menu containing the following options.



<b>Edit</b>	Edit the selected DATUM line definition. This option will display the CREATE/EDIT menu.
<b>Add to All Graphs</b>	Add the selected DATUM line definition to all the currently defined graphs
<b>Remove From All Graphs</b>	Remove the selected DATUM line definition from all the currently defined graphs
<b>Move Datum</b>	Make a copy of the selected DATUM line, the original definition will be deleted when the copy is inserted.
<b>Copy Datum</b>	Make a copy of the selected DATUM line.
<b>Insert Before</b>	Insert the previously copied/moved DATUM line definition before the selected DATUM line.
<b>Insert After</b>	Insert the previously copied/moved DATUM line definition after selected DATUM line.
<b>Delete</b>	This will delete the selected DATUM line.

### 5.24.2 Create/Edit

Each DATUM line must be defined with a unique acronym that is used to identify it in FAST-TCF scripts. The acronym shouldn't contain any spaces.

An optional label that is displayed on the graph next to the DATUM line can also be defined. As well as defining the font, size and colour for the label the position of the label relative to the DATUM line can also be set.

DATUM lines can be defined as

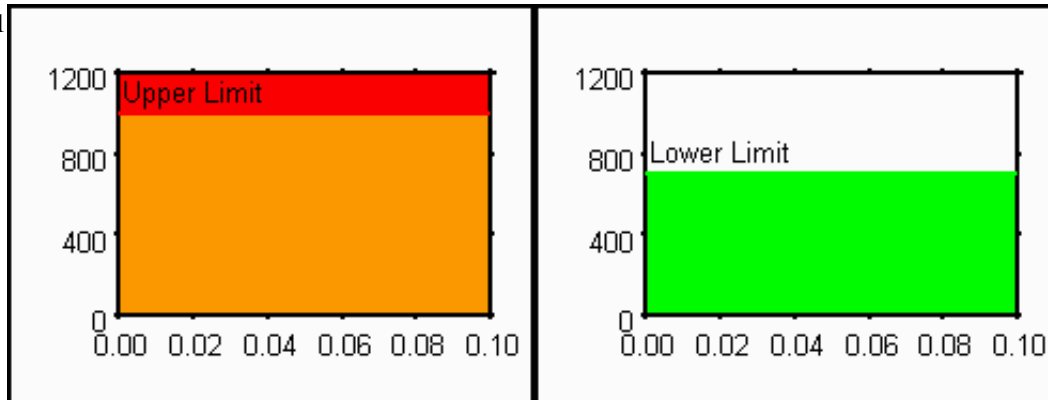
- Constant Y values
- Constant X values
- Curves of X,Y points

For a constant X or Y value the line will automatically extend to the edges of the graph and the areas either side of the line can be filled using any of the standard T/HIS colours.

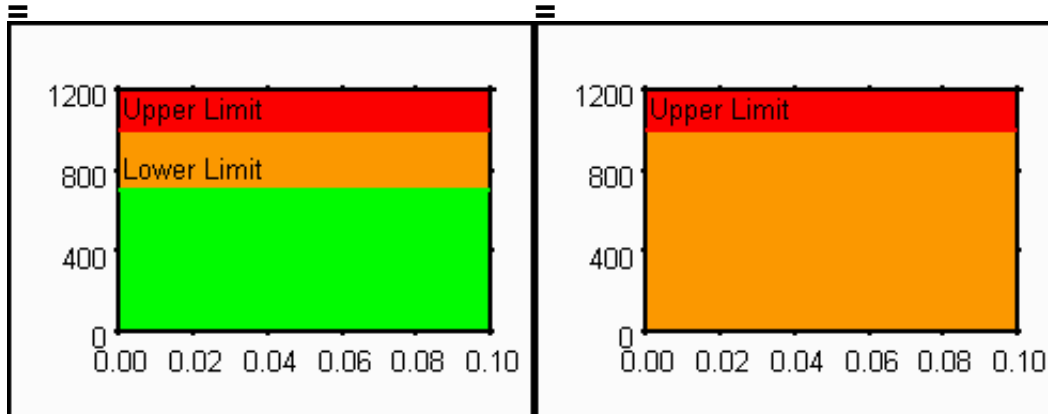
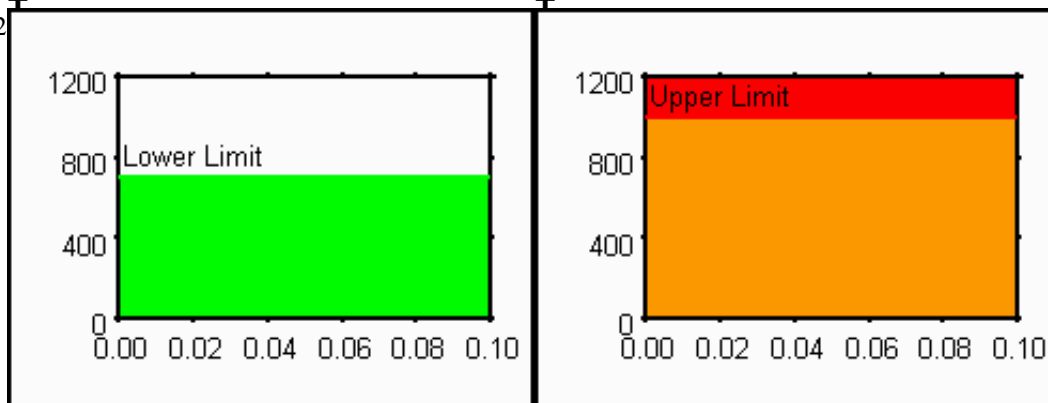
If a DATUM line is defined using X,Y points then the areas between the curve and the graph axis can also be filled.

As the DATUM lines are drawn in the order they are defined then care must be taken when applying fill colours

Datum 1



Datum 2

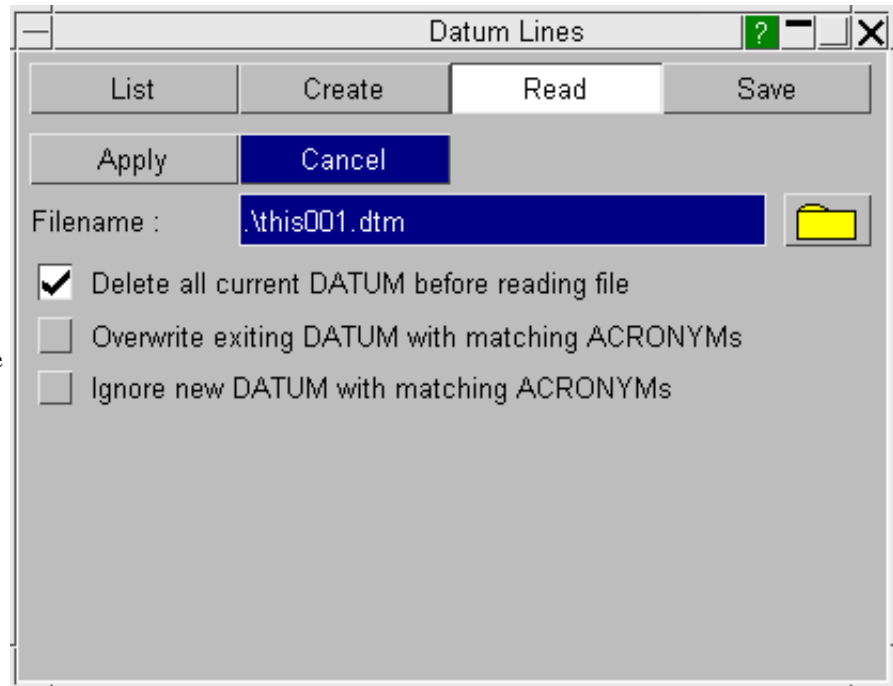


### 5.24.3 Read

This option can be used to read in a file containing DATUM line definitions that has previously been saved.

All DATUM lines must have a unique acronym. When the file is read the user had to the choice to

1. Delete any existing DATUM line definitions before the file is read.
2. If a DATUM line in the file being read has the same acronym as an existing DATUM line then the existing definition will be overwritten.
3. If a DATUM line in the file being read has the same acronym as an existing DATUM line then the new definition in the file will be ignored.



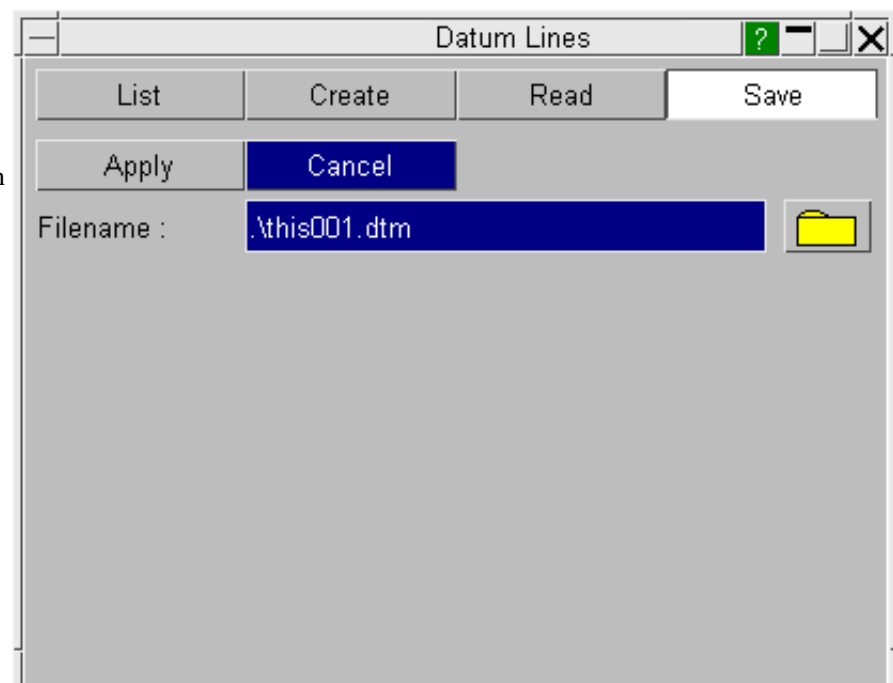
The preference option

**this\*datum\_file:** C:\datum\this001.dtm

can also be used to define a default file containing DATUM line definitions that is read automatically when T/HIS starts (see [Appendix H](#) for more details)

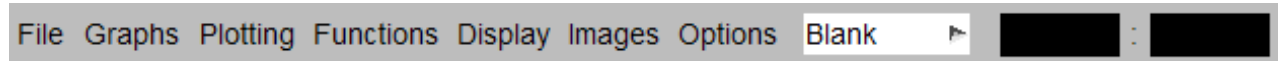
### 5.24.4 Save

This option can be used to save any DATUM line definitions to a file so that they can be reloaded and used in future T/HIS sessions.



## 6 Other Options

### 6.1 Tool Bar



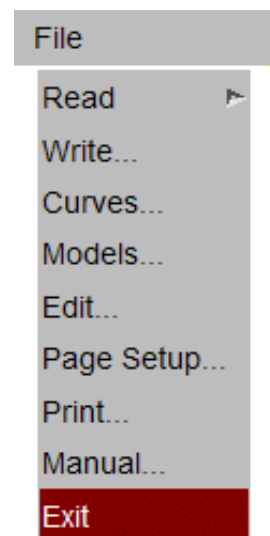
The tool bar is located across the top of the main T/HIS window and provides easy access to all of the main T/HIS menus from a series of drop down menus. In addition to the menus the drop down menus also allow a number of items to be changed dynamically and it provides a constant feedback of the cursor position within the graph area.

Each graph window contains it's own tool bar that provides a subset of the functions in the main toolbar ([see Section 6.2](#))

#### 6.1.1 File

The File drop down menu can be used to access the following menus.

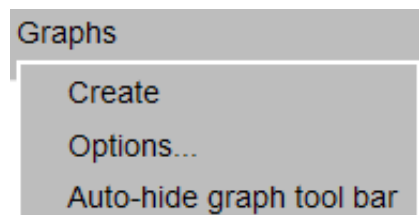
Read	see Section <a href="#">5.1</a> for more details.
Write	see Section <a href="#">5.2</a> for more details.
Curve Manager	see Section <a href="#">5.3</a> for more details.
Model Manager	see Section <a href="#">5.4</a> for more details.
Edit	see Section <a href="#">5.5</a> for more details
Page Setup	This option is only available on PC's and can be used to access the standard Windows Page Setup menu.
Print	This option is only available on PC's and can be used to access the standard Windows Print menu.
Manual	Displays this manual.



#### 6.1.2 Graphs

The Graphs drop down menu can be used to create new graphs and to change layout options.

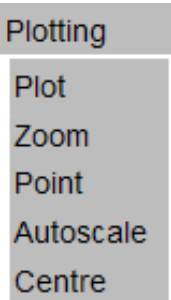
Create	Create a new graph, see Section <a href="#">3.1</a> for more details.
Options...	Modify graph layout options, see <a href="#">section 3.1</a> for more details.
Auto-hide graph tool bar	This option can be used to automatically hide the tool bar, see Section <a href="#">6.2</a> , at the top of each graph window.



### 6.1.3 Plotting

The Plotting drop down menu can be used to access the following plotting commands

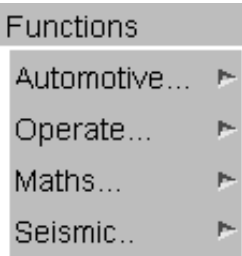
Plot	see Section <a href="#">4.1</a> for more details.
Zoom	see Section <a href="#">4.4</a> for more details.
Point	see Section <a href="#">4.2</a> for more details.
Autoscale	see Section <a href="#">4.5</a> for more details.
Centre	see Section <a href="#">4.6</a> for more details.



### 6.1.4 Functions

The Functions drop down menu can be used to access all of the curve functions.

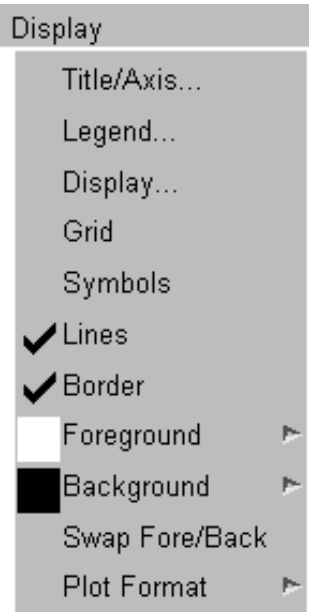
Automotive	see Section <a href="#">5.11</a> for more details.
Operate	see Section <a href="#">5.9</a> for more details.
Maths	see Section <a href="#">5.10</a> for more details.
Seismic	see Section <a href="#">5.12</a> for more details.



### 6.1.5 Display

The Display drop down menu can be used to access the Title/Axis and Display menus and to dynamically modify the appearance of graphs. This menu changes all of the currently active graphs ([see section 3.5](#))

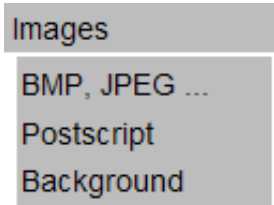
Title/Axis	see Section <a href="#">5.15</a> for more details.
Legend	see Section <a href="#">5.15.5</a> for more details.
Display	see Section <a href="#">5.16</a> for more details.
Grid	Turns the grid on/off, see Section <a href="#">5.16.4</a> for more details.
Symbols	Turns graph symbols on/off, see Section <a href="#">5.16.4</a> for more details.
Lines	Turns graph lines on/off, see Section <a href="#">5.16.1</a> for more details.
Border	Turns the plot border on/off, see Section <a href="#">5.16.6</a> for more details.
Foreground	Sets the foreground colour, see Section <a href="#">5.16.9</a> for more details.
Background	Sets the background colour, see Section <a href="#">5.16.8</a> for more details.
Swap Fore/Back	Swaps the current foreground and background colours, see Section <a href="#">5.16.10</a> for more details.
Plot Format	Set the current plot format, see Section <a href="#">5.15.5.2</a> for more details.



# 6.1.6 Images

The Images drop down menu can be used to save the current displayed graphs as an image in a number of formats. In addition to saving an image this menu can also be used to read in an image that is used as the background for each graph.

- BMP, JPEG... Capture the image as a bitmap or JPEG, see Section [5.8.1](#) for more details.
- Postscript Generate a Postscript or PDF image, see Section [5.8.2](#) for more details.
- Background This option can be used to set an image as the background for each graph, see Section [5.8.3](#) for more details.



## 6.1.7 Options

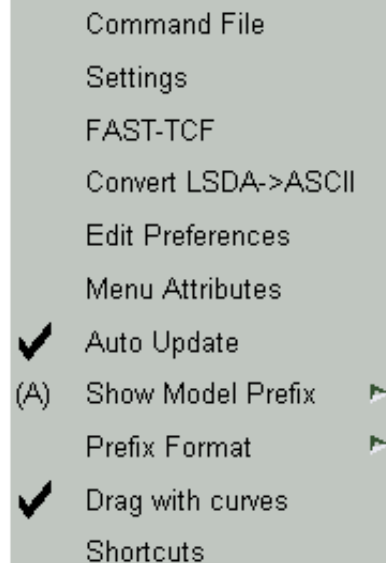
The Options drop down menu can be used to access all the following functions

Command File	see Section <a href="#">5.11</a> for more details.
Settings	Change data sources and other settings, see Section <a href="#">5.17</a> for more details.
FAST-TCF	Generate/playback FAST-TCF scripts, see Section <a href="#">5.10</a> for more details.
Convert LSDA>ASCII	Convert a LSDA binout file to ASCII, see Section <a href="#">5.4.4</a> for more details.
Edit Preferences	Displays the preference editor, see Section <a href="#">6.6</a> for more details.
Menu Attributes	Modify menu fonts, size and colours, see Section <a href="#">6.1.7.1</a> for more details.
Auto Update	Turn on/off automatic update.
Show Model Prefix	Turn the model prefix on/off or set it to automatic, see Section <a href="#">5.15.5.1</a> for more details.
Prefix Format	Select the prefix format displayed for each model. This option can be used to set the format used for the curve prefix. This option has 4 settings

<b>Model Number</b>	The model number will be used as the prefix. e.g. <b>(M1)</b>
<b>Directory</b>	The directory name the model was read from will be used at the prefix. e.g. <b>(run1)</b>
<b>THF File</b>	The root name of the THF file will be used as the prefix. e.g. <b>(sled_test)</b>
<b>User Defined</b>	A used defined prefix will be used. The prefix can be defined on a model by model case using the <a href="#">Model Menu</a> .

Drag with curves	Turn on/off the display of curves when dragging axis borders and legends. On some slow machines the time taken to update the display when a large number of curves is displayed makes the dragging response too slow. This option will automatically turn off the display of curves while the dragging operation is active.
Shortcuts	Setup keyboard shortcuts for commonly used function, see section <a href="#">6.5</a> for more details.

### Options





### 6.1.7.1 MENU Attributes

This panel allows you to tune the visual attributes of the screen menus within T/HIS and save them if you wish.

Menu Attributes

Dismiss Save\_Settings HELP

Display Factor: 1.30 0.5 (larger) .. 2.0 (smaller)

Font size:
 

☐ Small  
☐ Default  
☐ Large

Helvetica
 

☒ Permit scaling

CJK fonts
 

unix font: -misc-fixed-medium-r-normal-\*12-\*-\*-\*-\*-\*

windows font: MS Gothic 12

Brightness: 1.00 Menu brightness: 0.0 .. 1.0

Saturation: 1.00 Menu saturation: 0.0 .. 1.0

Gradation: 0.00 Button shade gradation: 0.0 .. 1.0

Left handed:
 

☐ None  
☐ Mouse  
☐ Shift & Ctrl  
☐ All

Left handed support swaps left and right mouse buttons and/or <shift> and <ctrl> keys or all of these

Dynamic viewing: Presets:
 

Synchronizing viewing: Icon + Caps lock

Meta key:	Shift key	Control key	Shift + Ctrl keys
Actions:	Current mode	Wireframe mode	Free-edge mode
Left mouse:	Rotation (XYZ)	Rotation (XYZ)	Rotation (XYZ)
Middle :	Translation	Translation	Translation
Right :	Zoom (Up +ve)	Zoom (Up +ve)	Zoom (Up +ve)

Scroll Factor:
 

1 Scroll %age 20  
 5

Sets the factor to zoom in and out by using the mouse wheel in graphics windows.

Zoom Factor:
 

1 Cursor %ag 20  
 5

Sets the factor to zoom in and out when using <shift> or <ctrl> + <right mouse>

3D Mouse Tuning

Rotn. factor: 1.00

Pan factor: 1.00

Zoom factor: 1.00

☐ MENU\_AUTO\_CONFIRM (affects this session only, setting is not saved)

## Display Factor

Lies in the range 0.5 to 2.0, default 1.0. Values < 1.0 reduce the apparent size of the screen so that menus and text become larger. Values > 1.0 act in the opposite sense. This is the simplest way of taking into account the display size.

## Font Size

Sets the font size independently of the display scale, face which can be useful on wide-screen displays. The font typeface can also be changed.

Font typeface
Helvetica Normal
Helvetica Bold
Times Normal
Times Bold
Courier Normal
Courier Bold

## Font scaling

By default text in menu interface buttons can be scaled downwards to a smaller font size (if one exists) if it is too long for the button. This shows more characters, but it can look messy when the user interface has a mixture of font sizes. Turning font scaling off prevents this happening, giving a more consistent appearance. (However it is generally better to adjust the Display Factor in order to find a menu scale that gives consistent font sizes.)

## Brightness

Lies in the range 0.0 to 1.0, default 1.0. Controls the brightness of the menu interface only (it will not affect displayed graphics).

## Saturation

Lies in the range 0.0 to 1.0, default 1.0. Controls the colour saturation of the menu interface. (Again it will not affect displayed graphics.)

## Left Handed

The software uses mouse buttons and keyboard 'meta settings keys (<shift> & <control>) in a handed way that is set up by default for right-handed use. It is possible to configure either or both for left-handed use.

## Save Settings

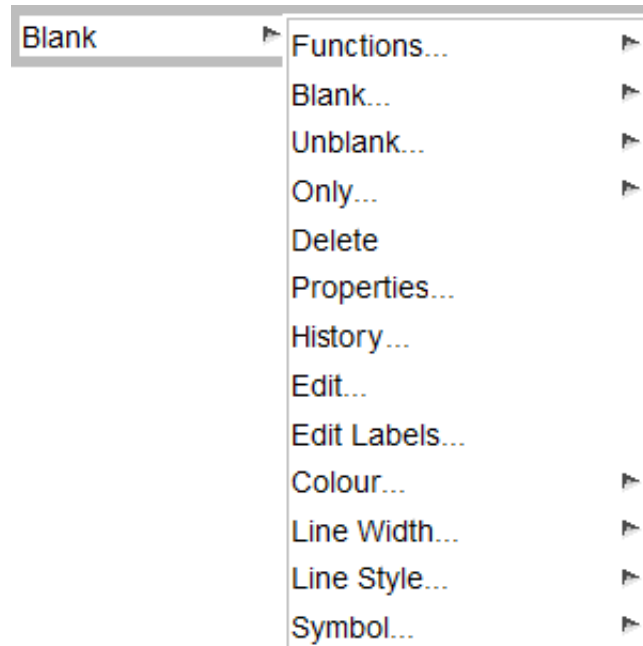
Once you have adjusted the above to your taste you can save these settings in your 'oa\_pref' file for future use with the **Save\_Settings** button. If you do not save settings they will be lost when this session exits.

## 6.1.8 Quick Pick

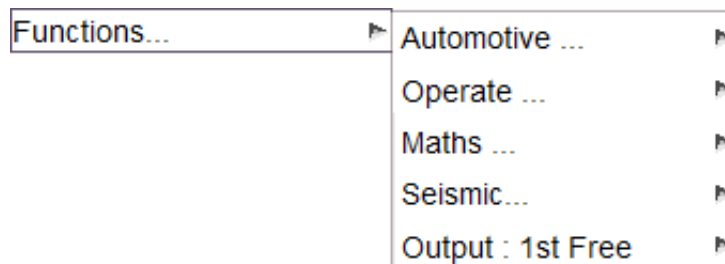
The Quick Pick menu can be used to perform many common curve operations using just the mouse. The current "Quick Pick" mode is displayed on the tool bar and can be changed using the popup menu.

The current "Quick Pick" option can be applied to a single curve by selecting the curve using the left mouse button. Multiple curves can be selected by holding down the left mouse button and dragging out an area.

Some functions can be undone using the middle mouse button.



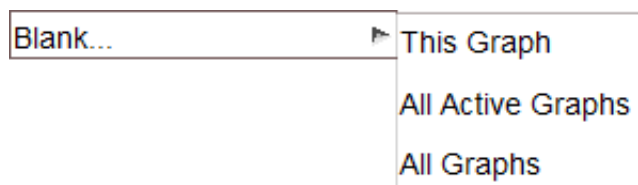
### 6.1.8.1 Functions...



This option can be used to select any of the curve operations (see Sections [5.9](#), [5.10](#), [5.11](#) and [5.12](#)) that have a single curve as input. In addition to selecting a curve operation this menu can also be used to set the output curve for the curve operation to either the 1st free curve or to overwrite the input curve.

This option can be applied to multiple curves but it can not be undone.

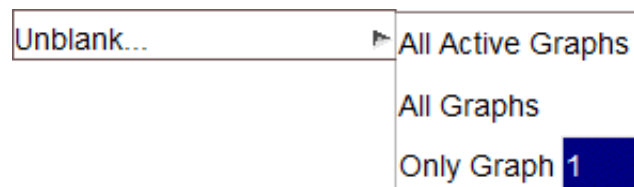
### 6.1.8.2 Blank...



This option can be used to blank curves. The selected curves can be blanked in just the graph they were selected in, all the currently active graphs or all graphs.

This option can be applied to multiple curves and it can be undone using the middle mouse button.

### 6.1.8.3 Unblank...



This option can be used to unblank curves. The selected curves can be unblanked in all the currently active graphs, all graphs or a individual graph can be specified.

This option can be applied to multiple curves and it can be undone using the middle mouse button.



This option can be used to blank all curves except for the selected ones. The selection can be applied to just the graph they were selected in, all the currently active graphs or all graphs.

This option can be applied to multiple curves and it can be undone using the middle mouse button.

#### 6.1.8.5 Delete

This option can be used to delete curves. It can be applied to multiple curves but it can not be undone.

#### 6.1.8.6 Properties...

This option will display the current properties for a curve (see Section [6.3.1](#) for more details). If multiple curves are selected this option is only applied to the one with the lowest curve ID.

#### 6.1.8.7 History...

This option can be used to view and edit the history of operations used to create a curve (see Section [6.4](#) for more details).

#### 6.1.8.8 Edit...

This option can be used to select a curve for editing (see Section [5.5](#) for more details). If multiple curves are selected this option is only applied to the one with the lowest curve ID.

#### 6.1.8.9 Edit Labels...

This option can be used to edit the label, title and axis labels for a curve (see Section [6.3.2](#) for more details) . If multiple curves are selected this option is only applied to the one with the lowest curve ID.

#### 6.1.8.10 Colours...

This option can be used to change the colour of curves. This option can be applied to multiple curves and it can be undone using the middle mouse button.

### 6.1.8.11 Line Width...

This option can be used to change the line width of curves. This option can be applied to multiple curves and it can be undone using the middle mouse button.

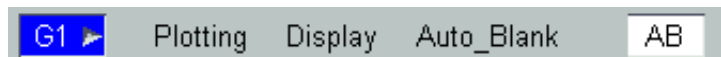
### 6.1.8.12 Line Style...

This option can be used to change the line style of curves. This option can be applied to multiple curves and it can be undone using the middle mouse button.

### 6.1.8.13 Symbols...

This option can be used to change the symbol style of curves. This option can be applied to multiple curves and it can be undone using the middle mouse button.

## 6.2 Graph Tool Bar



### 6.2.1 Graph Selection



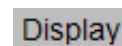
This option can be used to make a graph active or inactive, [see section 3.5](#) for more details.

### 6.2.2 Plotting



This option provided the same functions as the [Plotting](#) menu in the main toolbar with the exception that the settings only apply to the graph in the window instead of all of the currently active graphs.

### 6.2.3 Display



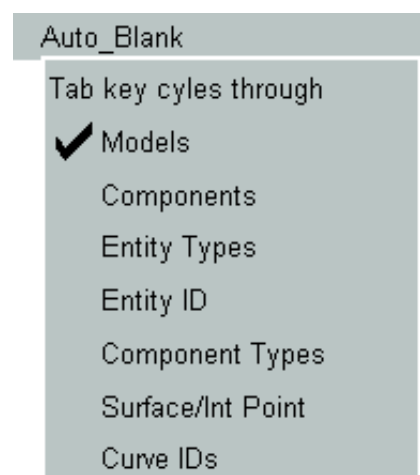
This option provided the same functions as the [Display](#) menu in the main toolbar with the exception that the settings only apply to the graph in the window instead of all of the currently active graphs.

### 6.2.4 Auto\_Blank

The **Auto\_Blank** function can be used to blank and unblank curves in a graph using either the TAB key or SHIFT+TAB.

By default if you now press the TAB key in a graph T/HIS will automatically blank all the curves except for those belonging to model 1. If you press TAB a 2nd time you will just see the curve belonging to model 2, a third time model 3. When you reach the end of the models you have curves for pressing the TAB key loops back to model 1. If you press SHIFT+TAB then it goes the other way (model 3 > model 2 > model 1 > model 3 ....)

Instead of blanking curves by model the behaviour of the TAB key can be changed.



Models	Default. Blanks curves by model ID.
Components	Blanks curves by component. e.g Node X Displacement > Node Y Displacement > Node Z Displacement > ...
Entity Types	Blanks curves by entity type e.g. Whole Model > Parts > Nodes > Solids > ...
Entity ID	Blanks curves by ID. e.g Node 1 and Solid 1 > Node 2 and Solid 2 > ....
Component Types	This is similar to Component except that it lumps all the displacement curves together then velocity so you get x,y,z and magnitudes. You will also get data for different entity types. So Energy would show things like Whole Model KE and Contact Energies.
Surface/Int Point	Blanks curves by surface or through thickness integration point. e.g Top > Middle > Bottom > Layer 1 > ...
Curves ID's	Blanks curves by ID

The default **Auto\_Blank** mode can be modified using the preference file (see [Appendix H](#) for more details)

this\*auto\_blank\_mode:

### 6.2.5 AB



This option can be used to turn on and off the Auto Blank option. The default setting for this option can be modified using the preference file (see [Appendix H](#) for more details)

this\*auto\_blank:

## 6.3 CURVE INFORMATION

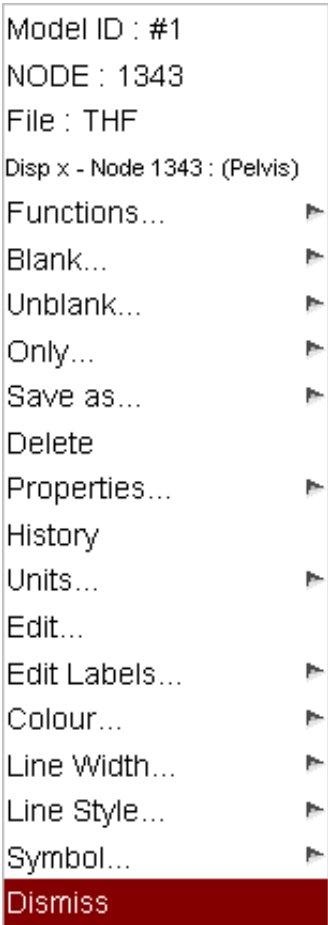
Pressing the right mouse button while in the graphics window will display a popup menu listing the ID, label and the data source of the nearest curve.

When data is read from either one of the LS-DYNA output files T/HIS will store the ID and type of the entity that the data applies to. If the curve label is modified this data will remain unchanged so that the curve source can still be identified.

If a curve has been read in from another source then T/HIS will report the data source as being UNKNOWN

If a curve is created from another curve using one of the T/HIS curve operations then the data source for the new curve will be copied from the original curve. If the operation uses more than one curve as input then the data source information will only be copied to the new curve if all of the input curves had the same data source.

**Edit** will open the curve editor for the selected curve whilst the colour, Line Width, Line style and Symbol pop-up menus allow the user to change these options for the curve (as can be done from the **STYLE** menu).



### 6.3.1 Properties...

This option displays a number of properties for a curve including minimum and maximum values, average and RMS value.

Xmin	0.0000000	
Xmax	0.0998993	
Ymin	0.0000000	@ X = 0.0000000
Ymax	1217170.8	@ X = 0.0479982
RMS	343394.38	
Average	271682.56	

### 6.3.2 Edit Labels...

This option can be used to change the title, tag, line label and axis labels for a curve.

Line Label	Accel mag - Node 1348
Title	LG09 : LARGE TEST 9: BELTED SLE
X-Axis	Time
Y-Axis	Acceleration
Tag	
<input type="button" value="Apply"/>	

### 6.3.3 Functions...

The functions popup menu can be used to access any of the curve operations that take a single curve as the only input. As well as applying an operation to a curve this menu can also be used to select between.

- Overwriting the input curve with the output from each function
- Writing the output to the 1st unused curve

Automotive ...	▼
Operate ...	▼
Maths ...	▼
Seismic...	▼
Output : 1st Free	▼

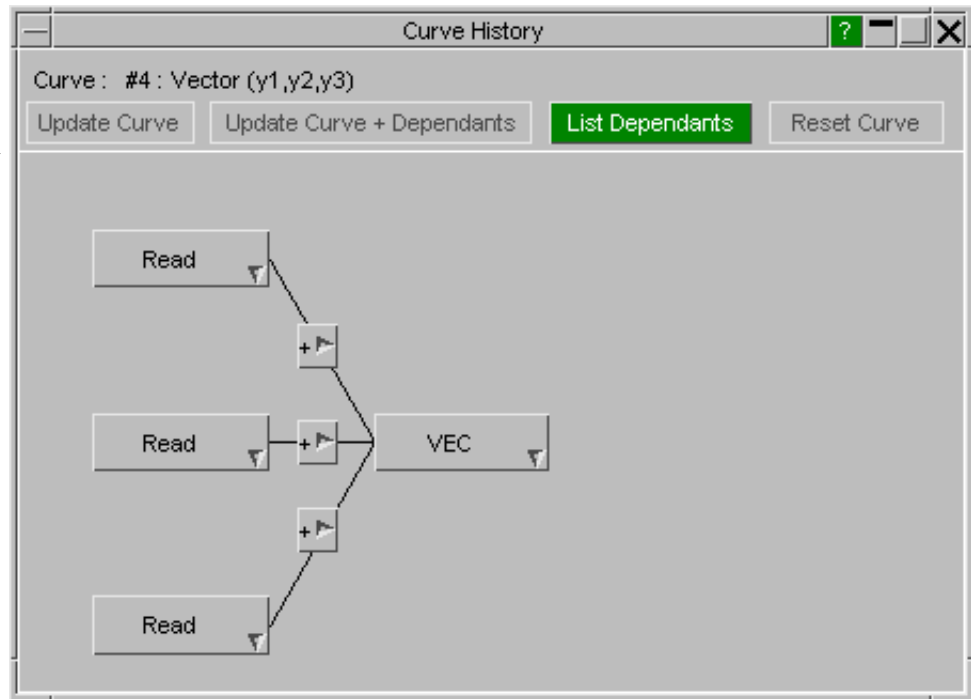
## 6.4 Curve Histories ...

Internally T/HIS knows about all of the operations used to create a curve and the order that the operations were applied. In addition to knowing the operations used to create each curve T/HIS also knows which curves were used as inputs to operations that created other curves.

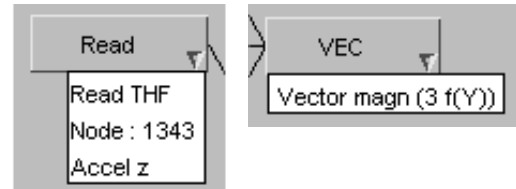
## 6.4.1 Viewing

When a curve is selected and the curve history is displayed a floating window will be displayed that shows all of the operations used to create a curve.

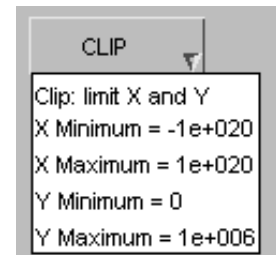
In the example opposite 3 items were read in and then combined using the **VEC**tor operation.



More information on each part of the curve history can be obtained by moving the mouse across each operation.



If a curve operation has one or more inputs that are not curves then the hover text will display all of the inputs along with their values.



## 6.4.2 Modifying

As well as viewing the operations used to create a curve the operations can also be modified by right clicking on them.

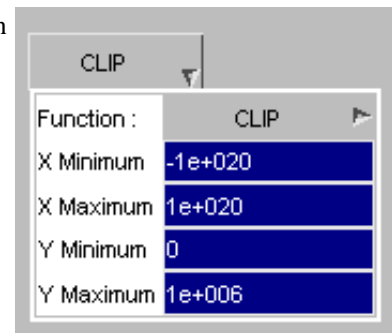
For a **READ** operation, the entity ID can be changed to any other ID of the same entity type. T/HIS will automatically check if results are available for the new ID and display a warning if they are not.

As well as changing the entity ID the data component can also be modified by selecting a different component in the popup menu.

Read			
Read :	THF	Disp x	RVel y
Type :	Node	Disp y	RVel z
ID :	1343	Disp z	RVel mag
Component :	Accel z	Disp mag	RAcc x
		Vel x	RAcc y
		Vel y	RAcc z
		Vel z	RAcc mag



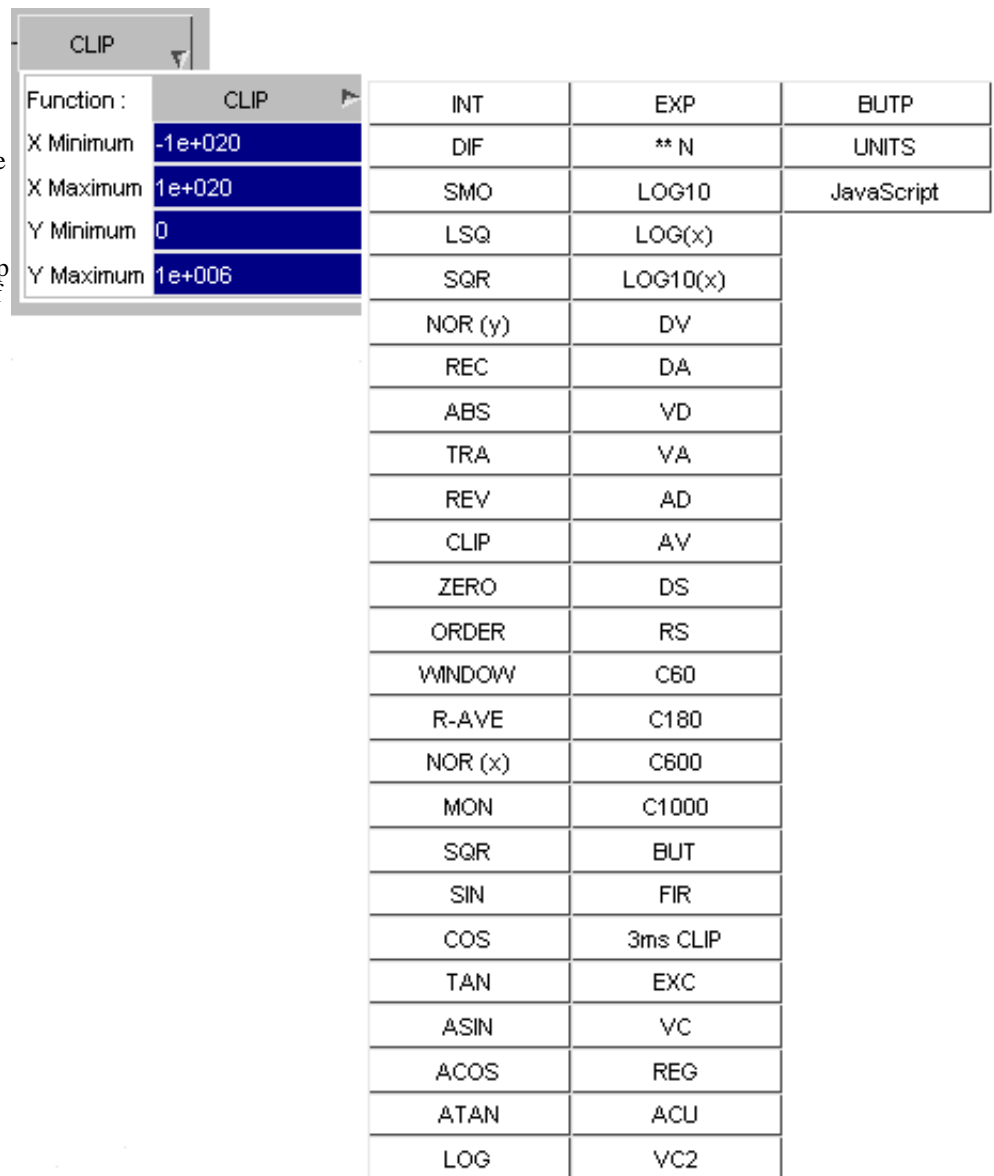
If a curve operation has one or more inputs that are not curves then right clicking on the operation will display a popup menu that will allow all of the values to be modified.



As well as changing the inputs to existing curve operations it is also possible to change a curve operation to any other curve operation that has the same number of input curves.

Right clicking on the popup symbol next to the name of the current curve operation will display a menu containing a list of all of the curve operations that are available which have the same number of input curves.

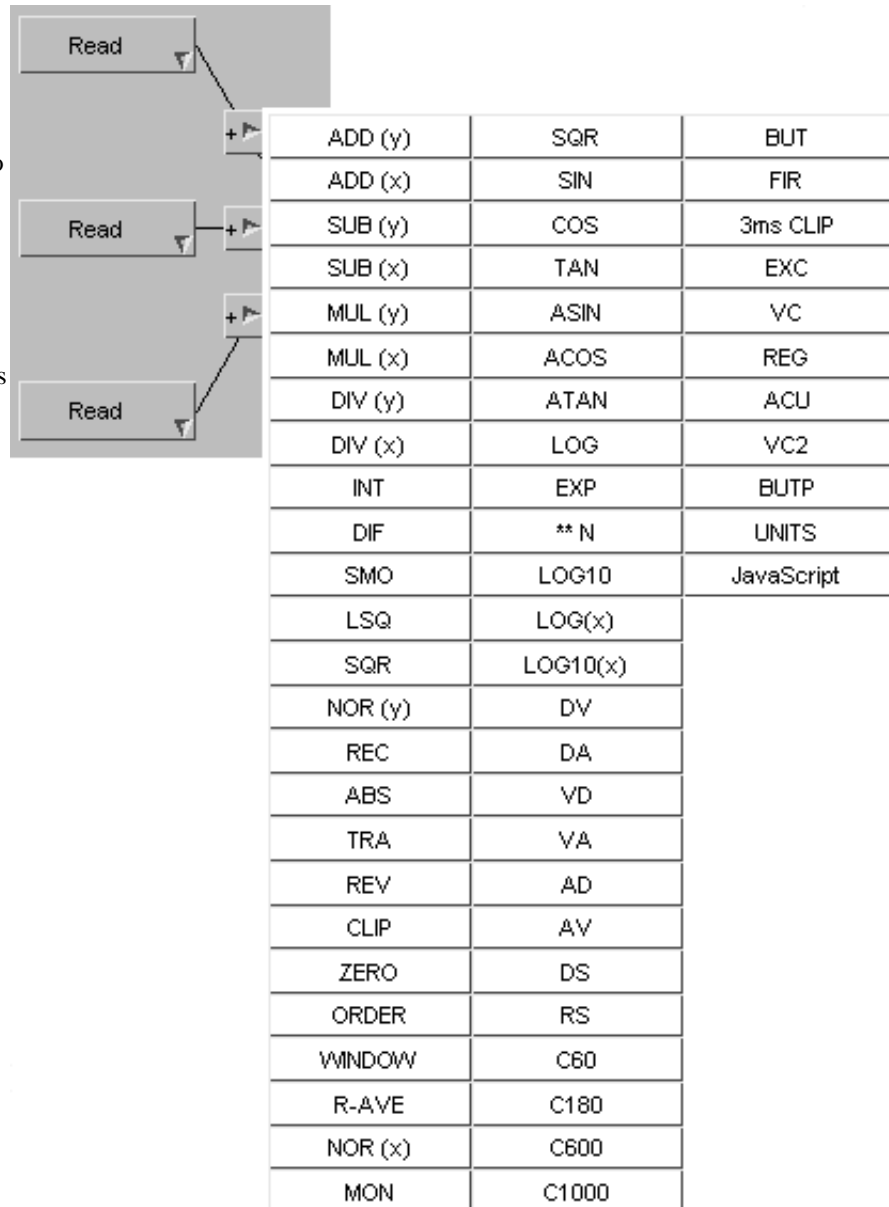
If for example the current curve operation is **CLIP** then the popup menu of available operations will contain all of the other curve functions that have a single input curve.



### 6.4.3 Inserting New Operations

New operations can be inserted into the chain of curve operations by right-clicking on one of the + symbols between the existing operations.

The popup menu that is displayed will contain all of the curve operations that take a single curve as input and produce a single output curve.



### 6.4.4 Update Curve

[Update Curve](#)

If any of the operations used to create a curve are modified or if a new operation is inserted then this option can be used to automatically update the curve. T/HIS will automatically rebuild the curve using the updated set of curve operations and will replace the old curve with the new one.

### 6.4.5 Update Curve + Dependants

[Update Curve + Dependants](#)

This option will update the selected curve and any dependant curves. As T/HIS stores all of the operations used to create every curve it knows if a curve has been used as an input to any other curves.

The selected curve will be automatically rebuilt and replaced with the new curve and then any curves that use the selected curve as an input will also be rebuilt and replaced.

### 6.4.6 List Dependants

[List Dependants](#)

This option will display a list containing any curves that have been created which use the currently selected curve as an input somewhere in their chain of curve operations.

### 6.4.7 Reset Curve

Reset Curve

This option can be used to reset all of the curve operations used to create a curve if any of them have been modified.

## 6.5 Keyboard Shortcuts

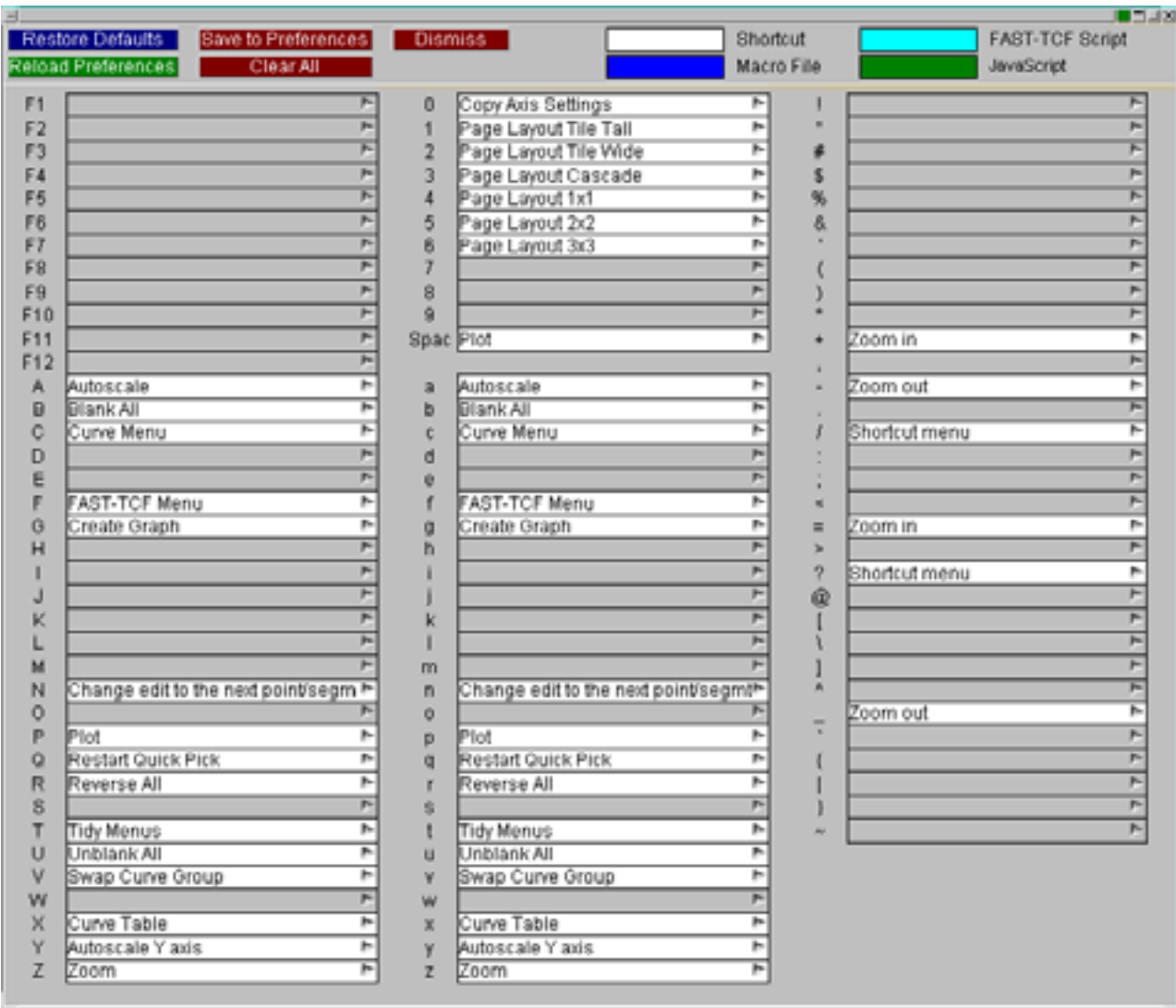
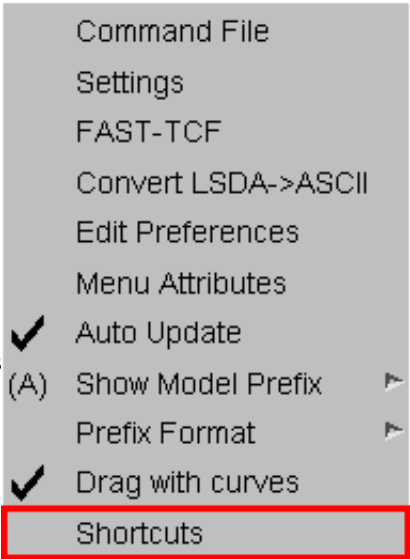
Some panels and actions can be accessed through pre-programmed shortcuts and from version 9.4 onwards the keys they are assigned to are customizable.

From version 9.4 onwards a number of new pre-programmed shortcuts have been added, including the top menu panels and window layout options. In addition to these pre-programmed shortcuts Macros and FAST-TCF scripts can also be assigned to a key.

A listing of the available shortcuts and the keys they are assigned to can be brought up by pressing the '?' key (by default) or accessing it through the Options top menu.

This will bring up a panel, from which you may assign the shortcuts, Macros and FAST-TCF scripts to the keys. Note that upper and lower case letters can be assigned different shortcuts.

A list of all the available pre-programmed shortcuts is given at the end of this section with their default key(s) if assigned.



At the top of the panel you will see the following buttons.

**Restore Defaults**

Restores the shortcuts to their default keys, removing any shortcuts assigned by the user.

**Save to Preferences**

Saves the shortcuts to the oa\_pref file in the home directory. They are saved in the format "this\*A\_key: AUTOSCALE" where the first part defines which key the shortcut is assigned to and the second part is the shortcut being assigned. Each shortcut has a specific name to use in the oa\_pref file, and a list is given below.

When T/HIS is started this is read and the saved shortcuts are restored.

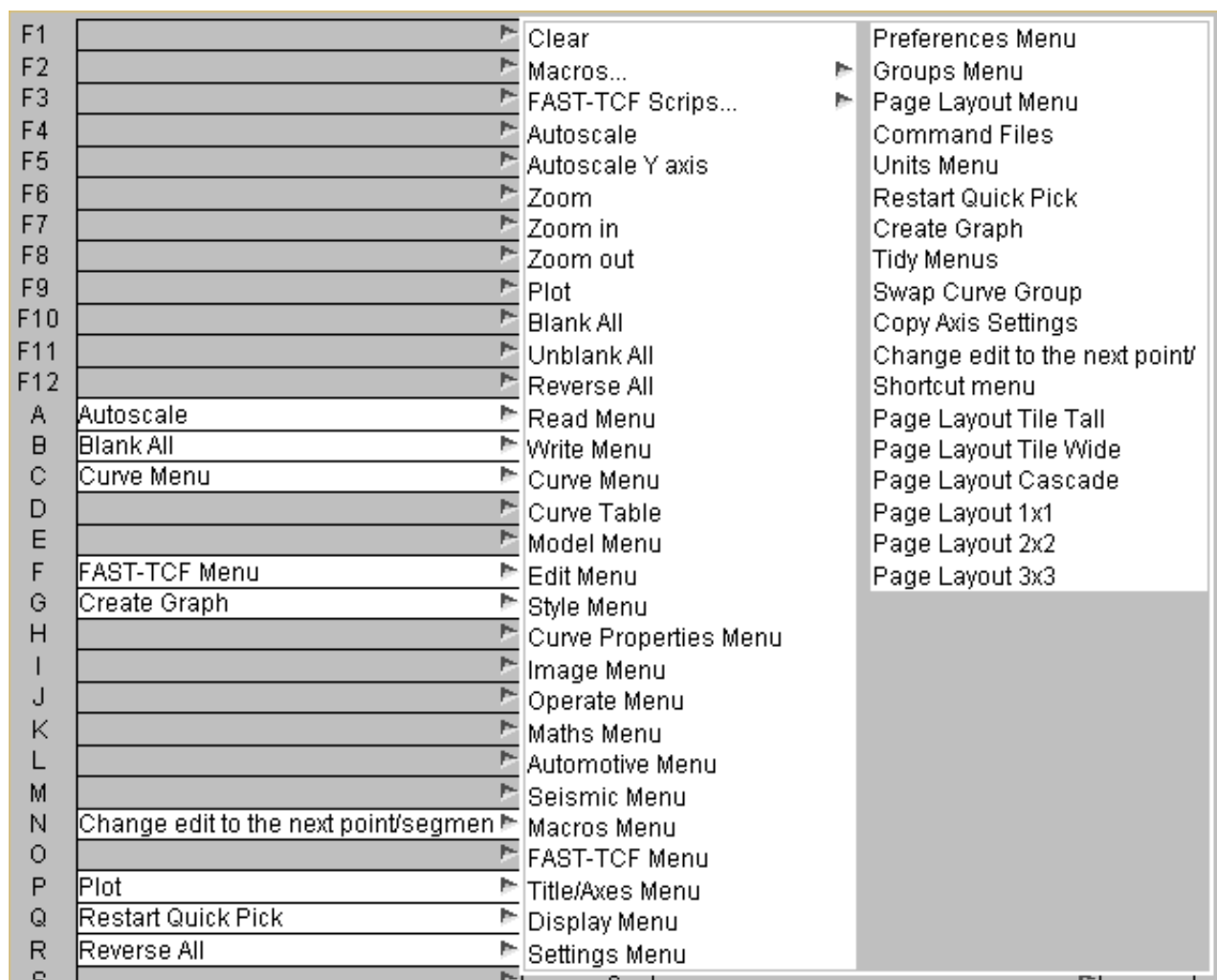
**Reload Preferences**

Reloads the shortcuts from the oa\_pref file in the home directory.

**Clear All**

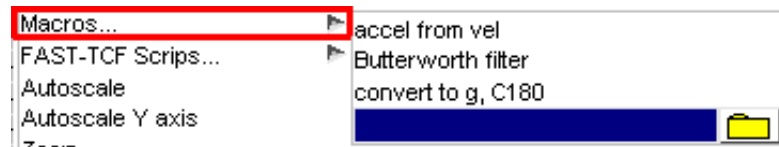
Clears all the shortcuts on the panel.

To assign a shortcut, right click on the key you want to assign it to. This will bring up a list of all available shortcuts in T/HIS as well as the option to assign Macros or FAST-TCF scripts.

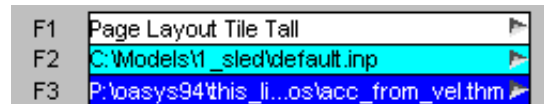


To assign a Macros, FAST-TCF script or JavaScript a to a key, right click on "Macros...", "FAST-TCF Scripts..." or "JavaScripts".

This will bring up another popup from which you can select the Macro or FAST-TCF script. The popup will contain a list of Scripts that T/HIS has picked up from the \$OASYS and home directory. If the script you want is not in this list you can browse for it by clicking on the folder icon.



The listing of assigned keys is colour coded to easily distinguish between pre-programmed shortcuts (white), FAST-TCF scripts (light-blue), Macros (dark-blue) and JavaScripts (dark-green)



**Pre-programmed Shortcuts:** Defaults shown in bold, oa\_pref name shown in brackets

<b>View Controls</b>	
<b>A/a</b> - Autoscale (AUTOSCALE)	Autoscale Y axis (Y_AUTOSCALE)
<b>P/p</b> - Plot (PLOT)	<b>[SPACE]</b> - Plot (PLOT)
<b>Z/z</b> - Zoom (ZOOM)	<b>"+" / "="</b> - Zoom in (ZOOM_IN)
<b>"-" / "_"</b> - Zoom out (ZOOM_OUT)	
<b>Blanking</b>	
<b>B/b</b> - Blank All (BLANK)	<b>R/r</b> - Reverse curve blanking (REVERSE)
<b>U/u</b> - Unblank all curves (UNBLANK)	
<b>Menus</b>	
Automotive Menu (AUTOMOTIVE_MENU)	Command Files Menu (CFILE_MENU)
<b>C/c</b> - Curve Menu (CURVE_MENU)	Curve Properties Menu (PROP_MENU)
Curve Table (CURVE_TABLE)	Display Menu (DISPLAY_MENU)
Edit Menu (EDIT_MENU)	Groups Menu (GROUPS_MENU)
Image Menu (IMAGE_MENU)	<b>F/f</b> - FAST-TCF Menu (FAST_TCF_MENU)
Macros Menu (MACROS_MENU)	Maths Menu (MATHS_MENU)
Model Menu (MODEL_MENU)	Operate Menu (OPERATE_MENU)
Page Layout Menu (PAGE_MENU)	Preferences Menu (PREF_MENU)
Read Menu (READ_MENU)	Shortcut Menu (SHORTCUT)
Seismic Menu (SEISMIC_MENU)	Settings Menu (SETTINGS_MENU)
Style Menu (STYLE_MENU)	Title/Axes Menu (TITLE_MENU)
Units Menu (UNITS_MENU)	Write Menu (WRITE_MENU)

Page Layout	
1 - Page Layout Tile Tall (TILE_TALL)	2 - Page Layout Tile Wide (TILE_WIDE)
3 - Page Layout Tile Cascade (CASCADE)	4 - Page Layout Tile 1x1 (LAYOUT_1X1)
5 - Page Layout Tile 2x2 (LAYOUT_2X2)	6 - Page Layout Tile 3x3 (LAYOUT_3X3)
Miscellaneous	
G/g - Create a new graph Window (NEW_WINDOW)	T/t - Tidy Menus (TIDY_MENUS)
V/v - Change Curve Picking Group (CURVE_GROUP)	Q/q - Swap to Quick Pick (QUICK_PICK)
PAGE UP - Next Page	PAGE DOWN - Previous Page
HOME - First Page	END - Last Page
Change edit to next point (EDIT_NEXT)	0 - Copy Axis Settings (COPY_AXIS)

## 6.6 Preferences

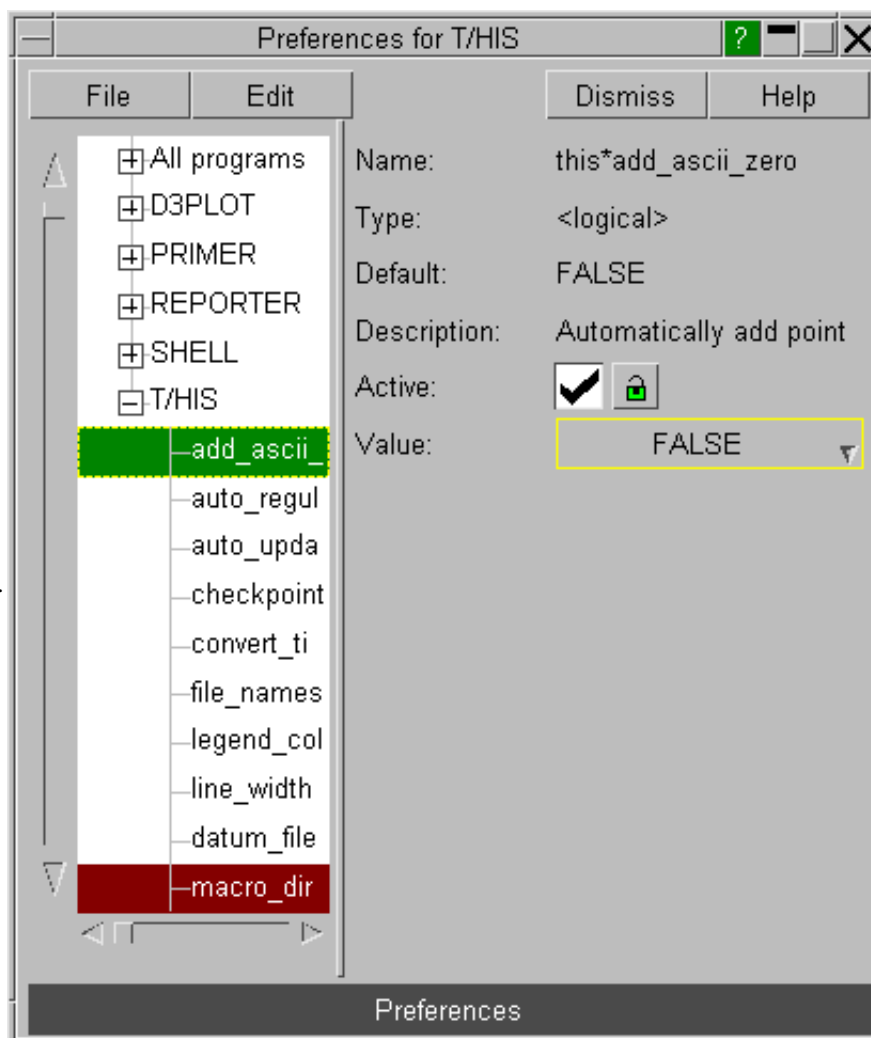
The Preference menu provides an interactive editor for setting options for T/HIS in the `oa_pref` preference file (see [Appendix H](#) for more details on the `oa_pref` file/options)

The preferences editor reads an XML file that contains all possible preferences and their valid options, and allows you to change them at will. In this example the user is changing the background colour in T/HIS.

Note that changes made in the Preferences editor will not affect the current session of T/HIS, they will only take effect the next time it is run.

If you have write permission on the `oa_pref` file in the `$OASYS` directory you will be asked if you want to update that file, otherwise you will only be given the option of updating your own file in your `$HOME` / `$USERPROFILE` directory.

For more information on the interactive preference editor see [Appendix H](#).







# 7 FAST-TCF

## FAST-TCF CONTENTS

- [7.0 Overview](#)
- [7.1 Introduction](#)
- [7.2 Page / Graph layout and selection](#)
- [7.3 Input syntax to load other files](#)
- [7.4 Input for data extraction requests](#)
- [7.5 Units](#)
- [7.6 Curve Tags](#)
- [7.7 Curve Groups](#)
- [7.8 Performing curve operations](#)
- [7.9 Applying extra options to data requests](#)
- [7.10 Setting properties for curves](#)
- [7.12 Image Output options](#)
- [7.13 Tabulation and presenter.var options](#)
- [7.14 FAST-TCF Curve Output](#)
- [7.15 FAST-TCF additional](#)

## 7.0 FAST-TCF OVERVIEW

FAST-TCF is a scripting language for T/HIS. It is designed to be editable and backward-compatible with previous versions of T/HIS. From version 9.2 FAST-TCF scripts can be recorded and played back in T/HIS. The FAST-TCF scripts are plain text files, and are therefore easy to edit and manipulate.

### 7.0.1 NEW FEATURES

#### New Features for FAST-TCF version 11.0

Version 11 of T/HIS contains the following new FAST-TCF commands

- [Support for DISBOUT data component](#)
- [Support for PLYOUT data components](#)
- ["style\\_m" command for setting curve styles by model](#)

#### New Features for FAST-TCF version 10.0

Version 10 of T/HIS contains the following new FAST-TCF commands

- [Support for TRHIST data components](#)
- [Support for CPM\\_SENSOR data components](#)
- [New wildcard options for specifying curve tags](#)
- [Outputting a range of curves to curve file](#)

#### New Features for FAST-TCF version 9.4

Version 9.4 of T/HIS contains the following new FAST-TCF commands

- [Support for DBFSI data components](#)
- [Support for TPRINT data components](#)
- [New "plot setup" commands](#)
- [New curve style options](#)

#### New Features for FAST-TCF version 9.3

Because of the multiple graphs and pages available in T/HIS 9.3 additional commands have been added to FAST-TCF 9.3 to define and position graphs and to generate multiple images containing one or more graphs. Because of these new commands version 9.3 FAST-TCF scripts generated by T/HIS can not be used in previous releases.

- [New commands have been added for generating and positioning multiple graphs and pages.](#)
- [New commands for generating images containing multiple graphs and pages.](#)
- [New variables have been added for accessing the output values of the ERR command.](#)
- [New built in variables "\\$run\\_nameN", "\\$run\\_titleN" and "\\$run\\_dirN" for multiple models.](#)

- New built in variable "\$FTCF\_PATH"

## New Features for FAST-TCF version 9.2

FAST-TCF has been extensively revised to include almost all of the T/HIS commands. The improved functionality does mean that old scripts may have to be changed to meet the new standards.

**NOTE: FAST-TCF is not 100% compatible with pre-version 9.1 input scripts:**

- Variables have changed to allow more flexibility, but the old rule for filenames (word1 + word2) has now been discontinued, filenames must all be one word
- Rigidwall command must now have "n" for the xtf file output (rather than nothing at all)
- Shell and Solid effective strain must have the fourth word "eff" to distinguish them from other types of strain that have been added
- No FAST-TCF defaults for plot setup - defaults are now the T/HIS standard ones

New features since version 9.1:

- Reading of keyword, csv, csv2, and bulk data files, keyboard entry
- Operation commands "order", "cat", "r\_ave", "stress", "logx", "logx10", "translate", "vector2D", "window"
- Variables are processed on a line by line basis
- Variables can be defined using curve properties - for example a variable could be set to equal max of a curve, and then used to divide another curve
- Continuation lines added - defined using a "\" at the very end of a line
- Tabulation commands "yatmax" and "yatmin" added for Y values at maximum and minimum X
- All extraction commands are supported: Boundary, Geo contacts, FSI, Joints, SPH, Thick shells and so on
- All the missing components for previous data types are now supported
- Multiple data extraction on one line e.g. "node 100:last acc X"
- Multiple generic tagging and labeling of output curves using wildcard "\*"
- Multiple curves can be operated upon in one line e.g. "oper ADD acc\_\* 10.0"
- Multiple curves can be plotted using wildcards "\*" in tag names
- Integration point output can be changed
- Multiple models supported
- Extended plotting syntax for setting up plot defaults (grid colours, offsets, fonts and so on)
- "Tabc" command for writing out tabulation data to a csv file
- "plot" and "auto" commands added for use in interactive playback mode
- macro support for running FAST-TCF files on specific curves

## 7.1 FAST-TCF INTRODUCTION

### 7.1.1 General Rules

1. **Each line** in the input file defines **one** data extraction or plot request
2. Long lines can be split into shorter ones using a continuation character "\" at the end of each line
3. **Space characters** are used to **divide the line into 'words'**
4. The input script is NOT case-sensitive.
5. Unless detailed elsewhere in this manual, the first few (usually three) characters of the first word on the line discriminate the request of a particular entity, and the syntax which applies to reading in the remaining words on the line
6. If the first word on the line is not recognised, the program ignores it - it is treated as a comment
7. The last words on the data extraction request lines allow [options](#) for filtering, Y-axis scaling, HIC, average and a short reference tag (The tags may be used for operation and plotting requests)
8. The last words on the plotting request line allow [options](#) for title, line style and axis changes
9. A successful data extraction always has a curve outputted, if there is no output (e.g. HIC, ERR) then a duplicate curve is outputted. This helps with tagging output curves

### 7.1.2 Running FAST-TCF

#### 7.1.2.1 Automatic running

FAST-TCF is integrated into the T/HIS executable and can be accessed from the command line or the shell.

**Command line syntax:**

<this executable> -tcf=<FAST-TCF input file> -start\_in=<start directory> -exit -batch <thf file name>

---

e.g. `this93.exe -tcf=side_impact.tcf -start_in=e:\side_impact\run1 -exit run1.thf`

The <thf file file>, -start\_in, -exit and -batch syntax are all optional.

**NOTES:**

- If no THF file is specified then T/HIS will search the directory for the latest one (\*.thf).
- If no THF file exists, then T/HIS will look for a d3thdt file (xtf file = xtf file).
- If this does not exist then no thf or xtf input filename is passed to FAST-TCF, and the input file is defaulted to ASCII
- The program runs in any directory you like (via the -start\_in command line option). The FAST-TCF output files are created in that directory, and files written out are relative to that directory.

Instead of opening a single model multiple models can be read using the command line option

<this executable> -tcf=<FAST-TCF input file> -start\_in=<start directory> -exit -batch -model\_list=<file name>

The -model\_list expects a text file with a list of filenames (1 per line) to read into model slots within T/HIS.

```
e.g   e:\side_impact\run1\run1.thf
      e:\side_impact\run2\run2.thf
      e:\side_impact\run3\run3.thf
      e:\side_impact\run4\run4.thf
```

**Shell operation:**

Right click on the SHELLS's T/HIS button, and go to the options menu. Select the FAST-TCF input script and the thf input file if necessary. Return to the main shell menu and press the T/HIS button.

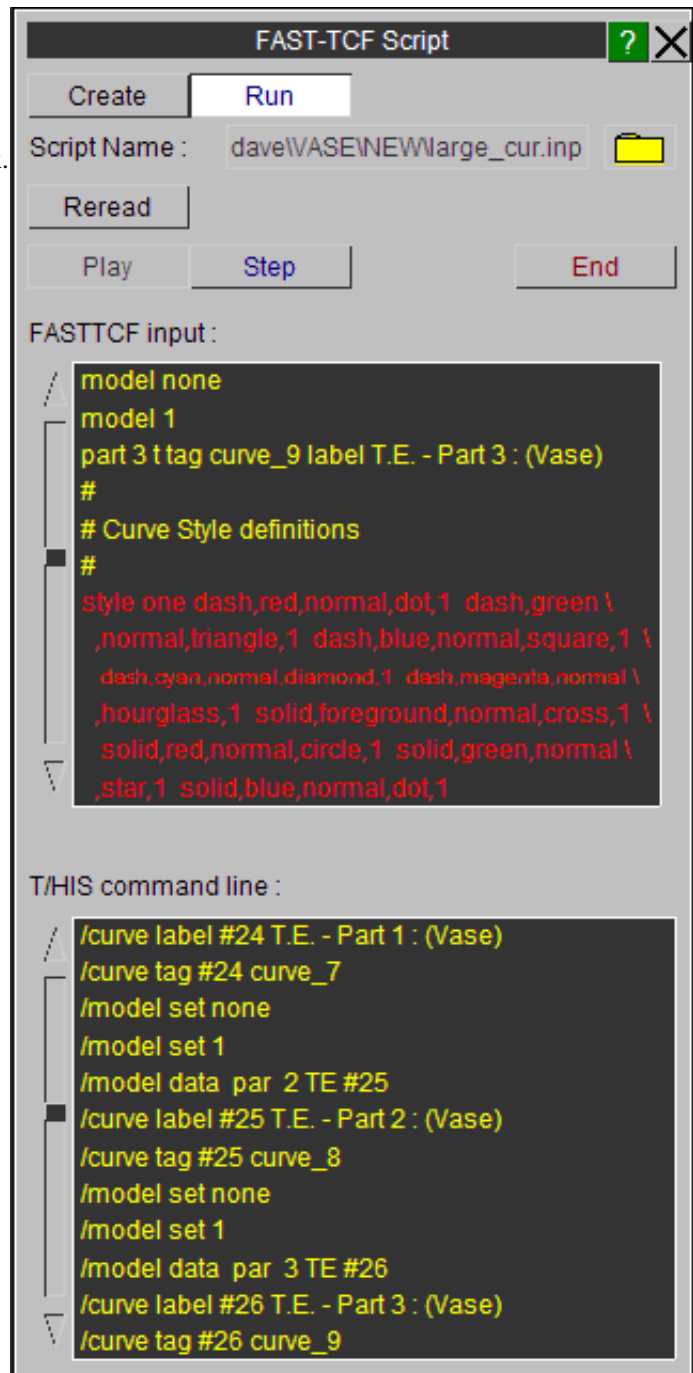
## 7.1.2.2 Interactive running

In the T/HIS tools menu within T/HIS, select the FAST-TCF option, then click on the "Run" tab in the sub menu that appears. This brings up the following menu:

The user can select the script file then with play the whole file through, or step through each command one by one.

The FAST-TCF line appears in the top dialogue box, and the translated T/HIS line appears in the bottom box. The line about to be sent to T/HIS appears in red text.

To end the script prematurely, hit the "End" button.



### 7.1.3 Input Files Needed, and Output and Intermediate Files Created

1. *input\_script* is **required** at the start.
2. *input\_script.output* is a file that contains the concatenated output from FAST-TCF.
3. *input\_script.tmp* is a temporary file that FAST-TCF creates for translation. This is merged after completion into *input\_script.output* so if you see this file then FAST-TCF didn't finish cleanly.
4. *input\_script.rep* is a temporary report file of the FAST-TCF run. This is merged after completion into *input\_script.output* so if you see this file then FAST-TCF didn't finish cleanly.
5. *input\_script.tcf* are the commands passed to T/HIS from FAST-TCF. This is merged after completion into *input\_script.output* so if you see this file then FAST-TCF didn't finish cleanly. The command lines contain special characters such as \r, \m and \l. These are used internally in T/HIS and should be ignored by the user.
6. *input\_script.sngval* contains summaries of every curve outputted.

Other files will be made, such as postscript or bitmap plots, but these will have names specified by the user.

## 7.1.4 Debugging FAST-TCF files

Complicated FAST-TCF files will inevitably go wrong. There are a number of things the user can do to help identify where it is going wrong. Assuming the command line syntax is correct and the correct files are in the run directory, these typical procedures are as follows:

### Identifying errors using the interactive playback option in T/HIS:

- Read the model(s) into T/HIS.
- Read the FAST-TCF script into T/HIS under the "FAST-TCF > Run" sub menu.
- Step through the FAST-TCF script manually, keeping an eye on how FAST-TCF is translating the lines, and the output T/HIS is producing.

### Identifying if FAST-TCF has found an error:

- If FAST-TCF finds an error, then it is stored and T/HIS then resets the command line and continues to translate the input file. If 10 errors are found then T/HIS will stop at this line. You can set this error amount internally within FAST-TCF.
- Once T/HIS has stopped, the errors are summarised in the command line box and the terminal that T/HIS was run from. The number of warnings found is also printed.
- It should be obvious what is wrong, FAST-TCF checks numerous things, including:
  - Whether T/HIS created the curve from the previous line.
  - That the syntax is correct for all the data input lines (the data extraction requests have additional checking to check the combinations of words inputted is right).
  - If the syntax is correct, whether it applies to the file being requested for output.
  - The output file exists in the directory for the data extraction.
- Correct the input line error utilising the reference tables in this document if applicable.

### Identifying what errors T/HIS is giving:

- Identify how many curves were outputted into T/HIS before things went wrong (run T/HIS in graphical mode).
- Place an exit keyword **after the next** input line. This should stop T/HIS just after the line which is causing the file to fail.
- Check what errors T/HIS is giving out. If it's not obvious what went wrong then try another procedure.

### Identifying if there are warnings or errors from FAST-TCF:

- The errors are summarised once T/HIS has finished. They are printed in the command line box and the terminal which T/HIS was run from.
- There will be a *input\_file.rep* or *input\_file.output* file in the directory which contains any warnings or errors that FAST-TCF has detected. Make sure nothing is obviously wrong with the input file using this report file.
- The *input\_file.tmp* or *input\_file.output* file contains the actual file inputted into FAST-TCF after includes have been found and special characters removed. Check this is correct and all the include files have been accounted for.

### Identifying if FAST-TCF is processing the line correctly:

- It's possible that FAST-TCF has processed the line incorrectly. If so, open the *input\_file.tcf* or *input\_file.output* file to investigate what FAST-TCF is asking T/HIS to do.
- Identify which line is going wrong using the [above procedure](#), and then find this section in the .tcf file. Input the entire tcf request for the line into the T/HIS command box to step through what is being asked from T/HIS. This may highlight where things are going wrong. The command lines contain special characters such as \r, \m and \l. These are used internally in T/HIS and should be ignored by the user.

### Using Primer to check a FAST-TCF file:

- Primer has a FAST-TCF check menu under the main check menu. This can be used to check the FAST-TCF file data requests against a certain keyword deck.
- Read the deck into Primer, and select MODEL > CHECK > CHECK FAST-TCF FILE. Select the FAST-TCF file and press APPLY. Details can be found in section 3.9 of the Primer manual.
- Primer will highlight any errors that have occurred with the input file with regards to the keyword deck.
- The main Primer checks are if the line syntax is valid, whether the correct file is being outputted, whether the relevant DATABASE\_HISTORY is present and whether the id. actually exists.
- Any errors will have to be corrected manually in Primer.

**NOTE:** If FAST-TCF has completed, then it may be necessary to open the *input\_file.output* file which has the all the output files concatenated together in different sections.

## 7.1.5 Creating FAST-TCF files

The most obvious option is to generate a FAST-TCF script using a text editor such as vim or wordpad. However, an easier option is to use T/HIS as normal, then generate a FAST-TCF script to recreate the curves currently displayed on the screen from within T/HIS.

**It involves a single button click to produce a FAST-TCF script that can recreate the plot on the screen.**

T/HIS internally stores the history behind each curve; noting which curves, operations and data requests were used to create each curve. This means that the user **does not** have to start recording a command file, and carefully record a script. Instead the user can work for as long as they like as normal, then choose to generate a FAST-TCF script to recreate the plot on the screen by using the FAST-TCF > Create menu.

By default the FAST-TCF script that is generated will contain commands to reproduce all of the graphs that are currently defined in T/HIS. Instead of reproducing all of the graphs the FAST-TCF script can also contain the commands to generate a subset or pages or graphs.

**FAST-TCF Script** [?] [X]

**Create** **Run**

**Apply**

Script Name :  [Folder Icon]

Generate for

- ☒ All Pages
- ☐ Current Page
- ☐ Only Page
- ☐ All Graphs
- ☐ All Active Graphs
- ☐ Only Graph

---

FASTTCF Script : Image Output ☒

Format :  [Dropdown Arrow]

Filename :  [Folder Icon]

---

FASTTCF Script : Curve Output

Filename :  [Folder Icon]

- ☐ Unblanked Curves
- ☐ Select Curves [...]

---

FASTTCF Script : Curve Group Output ☒

- ☒ All Curve Groups
- ☐ Select Curve Groups [...]

### 7.1.5.1 FAST-TCF Script : Image Output

This option can be used to add the commands to the FAST-TCF script to generate an image of each graph/page that is selected for output. In addition to selecting the image format a filename can also be specified that is used in the FAST-TCF script as the output filename for images.

### 7.1.5.2 FAST-TCF Script : Curve Output

This option can be used to add commands to the FAST-TCF script to write curves out to a T/HIS curve file. By default this option will add commands to the FAST-TCF script write any curves that are unblanked in a graph to a curve file. Instead of writing all of the unblanked curves out to a file the "Select Curves" option can be used to select a subset of curves.

### 7.1.5.3 FAST-TCF Script : Curve Group Output

This option can be used to select additional curves for output to the FAST-TCF script by curve group. If a curve is selected that is also unblanked in one of the graphs the command to regenerate it are only added to the FAST-TCF script once. This option will also add the commands to regenerate the selected curve groups to the FAST-TCF script.

## 7.2 PAGE / GRAPH LAYOUT AND SELECTION

FAST-TCF scripts can contain commands to create and position multiple graphs. T/HIS Pages can also be created and graphs moved between pages. By default T/HIS will automatically create a single graph on the 1st 'Page' when it starts. If a single graph is required then the script does not need to contain any of the commands in this section. If additional graphs are required then by default they will be created on the 1st Page unless multiple pages have been selected.

Keyword	2nd word	3rd word	4th word	5th word	6th word	7th word	8th word	9th word	notes
Layout	page	wide	-	-	-	-	-	-	Set the page layout to tile wide
		tall	-	-	-	-	-	-	Set the page layout to tile tall
		cascade	-	-	-	-	-	-	Set the page layout to cascade
		1x1	-	-	-	-	-	-	Set the page layout to 1 by 1 graphs per page
		2x2	-	-	-	-	-	-	Set the page layout to 2 by 2 graphs per page
		3x3	-	-	-	-	-	-	Set the page layout to 3 by 3 graphs per page
		XY	m	n	-	-	-	-	Set the page layout to (m) by (n) graphs per page
		custom	-	-	-	-	-	-	Set the page layout to custom
		n	all	-	-	-	-	-	Add all graphs to page (n)
		n	none	-	-	-	-	-	Remove all graphs from page (n)
		n	add	graph	ID	-	-	-	Add graph (ID) to page (n)
		n	remove	graph	ID	-	-	-	Remove graph (ID) from page (n)
		size	m	n	-	-	-	-	Set the page size to m by n pixels
		size	auto	-	-	-	-	-	Set the page size to automatic
	graph	total	n	-	-	-	-	-	Set the total number of graphs to (n)
		create	-	-	-	-	-	-	Create a new graph
		delete	all	-	-	-	-	-	Deletes all graphs except the first one.
		delete	n	-	-	-	-	-	Delete graph (n)
		position	n	x1,y1	x2,y2	-	-	-	Position graph (n) with the bottom left hand corner at screen location (x1,y1) and the top right hand corner at (x2,y2). All coordinates should be in the range 0.0 to 1.0.
		select	all	-	-	-	-	-	Select all graphs
		select	n	-	-	-	-	-	Select graph (n)
		select	none	-	-	-	-	-	Deselect all graphs
		n	axes	position	left	right	top	bottom	Set the position of the left, right, top and bottom axis for graph (n). The positions given should be in the range 0.0 to 1.0 or the word 'Auto'
		n	legend	position	left	right	top	bottom	Set the position of the left, right, top and bottom of the legend for graph (n). The positions given should be in the range 0.0 to 1.0 or the word 'Auto'
		n	legend	format	<type>	-	-	-	Set the legend format to one of <i>column/default, full/off, automatic, floating</i> for graph (n)
		n	legend	columns	n	-	-	-	Set the number of columns in the legend to n (1 to 3)
		n	legend	background	standard colour	-	-	-	Set a background colour for the floating legend
		n	legend	transparency	integer (0-100)	-	-	-	Set the background transparency for the floating legend
		n	x	format	<type>	-	-	-	Set the x axis unit format to one of <i>automatic, general, scientific</i> for graph (n)
		n	y	format	<type>	-	-	-	Set the y axis unit format to one of <i>automatic, general, scientific</i> for graph (n)
		n	y2	format	<type>	-	-	-	Set the second y axis unit format to one of <i>automatic, general, scientific</i> for graph (n)
		n	x	precision	m	-	-	-	Set the number of decimal places displayed for the x axis values to (m) in graph (n)
		n	y	precision	m	-	-	-	Set the number of decimal places displayed for the y axis values to (m) in graph (n)
		n	y2	precision	m	-	-	-	Set the number of decimal places displayed for the second y axis values to (m) in graph (n)



## 7.3 INPUT SYNTAX TO LOAD OTHER FILES

FAST-TCF has the option of reading in curve files and other FAST-TCF files nested within the input file. T/HIS now writes out and reads in curve styles and internal tags. FAST-TCF recognizes these tags if the user wishes to refer to them later on in the input file. If they are relative then the include files must be relative to where T/HIS is running from.

Filenames can contain spaces, but if they do then they **must** be enclosed in quotes

e.g. read "c:\my documents\filename.cur".

Description	keyword	second word	third word onwards	notes
Bulk data	readb	bulk data file	-	curves will be read in at this point in the file, and will be numbered accordingly
CSV 1 (X,Y,X,Y...)	readcsv	csv file	lr <row number containing line labels> ar <row number containing axis labels>	Subsequent words can be any of these 2 options. If no options then assumes reading x from column 1 and no labels.
CSV 2 (X,Y,Y,Y...)	readcsv2	csv file type 2	xg <x start value> <x interval> xc <x values column number> lr <row number containing line labels> ar <row number containing axis labels>	Subsequent words can be any of the 3rd word options. Only one of the options XG and XC can be used. If no options then assumes reading x from column 1 and no labels.
T/HIS Curve file	rea	curve name	-	curves will be read in at this point in the file, and will be numbered accordingly curve tags and styles are stored automatically through the \$TAG and \$STYLE lines NOTE: If the tag in the curve file conflicts with an existing tag, the tag is NOT read in
Keyword	readk	keyword file name	-	curves will be read in at this point in the file, and will be numbered accordingly
FAST-TCF Include	inc	include file name	-	FAST-TCF will search for includes within includes etc FAST-TCF pastes the include files into the final input file as soon as they are detected
LS-PrePost Curve file	readlspost	filename	-	Reads in curves from an LS-PREPOST curve file
LS-PrePost XY data file	readlsp_xy	filename	-	Reads in curves from an LS-PREPOST XY data file
DIAdem	read_diadem	header file filename	channel number to read	subsequent words can be either of these 2 options xg <x start value> <x interval> xc <channel containing x-axis> Only one of these 2 options can be used
JavaScript	java	JavaScript file name	-	Runs a JavaScript. If any curves created by the JavaScript are referenced by following command in the FAST-TCF script then the JavaScript should generate curves tags for the curves which can then be used in the FAST-TCF script.

Keyboard entry can also be added into the FAST-TCF file, allowing for simple curves to be created in T/HIS. The keyword for this is **keyboard**. The order of the following words is important, and must be adhered to (see below). The continuation line character is useful here "\".

Keyword	following word	following word	notes
Keyboard	xaxis	x axis name	specifies the x axis label
	yaxis	y axis name	specifies the y axis label
	label	curve label	specifies the curve label
	data	xval,yval xval2,yval2 xval3, yval3 etc	no space between the x and y values, only a space between the <b>pairs</b> of values

for example, to create an acceleration curve with a straight line at value 1.0:

**keyboard title straight line \ xaxis time \ yaxis accn \ label straight line at 1.0 \ data 0.000000,1.000000 \ 1.000000,1.000000**

## 7.4 INPUT FOR DATA EXTRACTION REQUESTS

Each data extraction request occupies one line, with the 'words' on the line separated by space characters.

The line starts with a keyword and the required arguments follow, then any optional requests can occur after the arguments (see later on in the manual).

ID can be a number **or a name** (enclosed in quotes ""), depending on whether the LS-DYNA version supports it in the relevant output file.

### Multiple data requests

T/HIS 9.2 onwards supports multiple data output syntax. T/HIS will read the data in one file pass, making it much quicker for larger runs. To use this in FAST-TCF you need to specify the range using a colon (:) and it must be in a single word. As well as the standard numbers you can use, there are some special words namely "all", "first" and "last" (see example).

<b>e.g.</b>	<b>whole_model</b>	<b>te</b>	<b>lsda</b>
	(whole model)	(total energy)	(force lsda file)
	<b>node</b>	<b>42</b>	<b>force</b> <b>y_dir</b>
	(node extraction)	(i.d. 42)	(force in y-direction)
	<b>node</b>	<b>"end of roof"</b>	<b>accel</b> <b>z</b>
	(node extraction)	(i.d. "end of roof")	(z acceleration)
	<b>node</b>	<b>100:last</b>	<b>force</b> <b>y_dir</b>
	(node extraction)	all nodes from 100	(force in y-direction)
	<b>node</b>	<b>all</b>	<b>force</b> <b>y_dir</b>
	(node extraction)	all nodes	(force in y-direction)

### 7.4.1 Selecting Models

If T/HIS contains more than one model the data extraction commands will attempt to read data from all the model that are currently selected. To specify which model to read data from the following commands can be used

Keyword	second word	notes
model	n	Select model "n" for reading data from
	all	Select all models for reading data from
	none	Unselect all models

### 7.4.2 Data Extraction options

#### 7.4.2.1 Specifying Files for data extraction

For some LS-DYNA data types results can be extracted from multiple files. By default FAST-TCF scripts will extract data from the default T/HIS file type for each entity type ([see Section 5.17.1](#)). These defaults can be changed via the [preference file](#).

Instead of using the default file any of the valid files types can be specified by using either the [define file](#) keyword (e.g. define file LSDA) or by adding an [extra line option](#). When this occurs, FAST-TCF will take the extraction request from the specified type of file - **but only if T/HIS allows it**.

Keyword	second word	third word	notes
define	file	lsda	will always check that t/his can get the output from this file, if not then the original default file will be chosen (see data extraction table). This file can still be overwritten on the actual input line
		ascii	
		xtf	
		thf	
		default	

e.g. `node 42 displacement x`  
 (read data from default file)  
`define file LSDA`  
`node 42 displacement x`  
 (read data from LSDA file)  
`node 42 displacement x ASCII`  
 (read data from ASCII file)

### 7.4.2.2 Specifying components for Steady State Dynamics (SSD) analysis

For a SSD analysis LS-DYNA generates 2 data values, an amplitude and an angle, for each component in the NODOUT and ELOUT parts of the LSDA (binout) file. By default FAST-TCF will extract the amplitude for each data component but this can be changed if required to extract the angle value

Keyword	second word	third word	notes
define	ssd_comp	amplitude	selects the amplitude value for all following data requests
		angle	selects the angle value for all following data requests

e.g. `define ssd_comp angle`  
 (read angle value for all SSD analysis data components)  
`define ssd_comp amplitude`  
 (read amplitude value for all SSD analysis data components)

### 7.4.3 Defining Groups of Parts

Description	keyword	second word	following words
Group definition	gdef	group id	part ids

The line starts with 'gdef' and is followed by an integer for the group i.d, and then part i.d. numbers separated by spaces, or for a range of parts - separated by a ':'.

- No options should be applied to this card, because all the words on the line are written out as integers.
- The input is on one line (which may result in a long line ...).

e.g. `gdef 1 1 2 3 4 10:20 30:40`  
 (group define i.d. 1) (parts 1 2 3 and 4) (parts 10000 to 20000 and 30000 to 40000)

### 7.4.4 Specifying Surfaces, Integration Points and Nodal Locations for data extraction

#### 7.4.4.1 Specifying Surfaces and Integration Points

From version 12.0 onwards the syntax for specifying which surface or integration point to read data from for Shells, Thick Shells and Beams has changed. These options are now appended to data extraction as follows.

## Shells and Thick Shells

extra word #1	extra word #2	notes
surface	top	If no surface option is specified then the default (middle) surface will be used.
	middle	
	bottom	
	n	

e.g. `shell 99 stress xx tag curve_1`

(read x stress for shell 99 middle surface)

`shell 99 stress xx surface top tag curve_1`

(read x stress for shell 99 top surface)

`shell 99 stress xx surface 3 tag curve_1`

(read x stress for shell 99 layer 3)

## Beams

extra word #1	extra word #2	notes
ipoint	n	Specifies the beam integration point to read data from

e.g. `beam 99 stress x ipoint 1 tag curve_1`

(read axial stress for beam 99 integration point 1)

### 7.4.4.2 Specifying in-plane integration points for Shells and Thick Shells

In recent versions of LS-DYNA it is possible to write out data at multiple in-plane integration points for fully integrated Shells and Thick Shells for each through thickness layer.

For fully integrated solid elements data can also be written out for all 8 integration points.

By default T/HIS will automatically read the average value for each element. If the element isn't fully integrated then the data for the 1st point will be used, if it is fully integrated and has multiple integration points then the average value will be calculated.

extra word #1	extra word #2	notes
ipoint	n	Specifies the in-plane integration point to read data from.  If this option isn't specified then the surface centre value will be selected. If the element is fully integrated then the average value will be calculated from all 4 in-plane values

e.g. `shell 99 stress xx tag curve_1`

(read x stress for shell 99 middle surface, centre value)

`shell 99 stress xx ipoint 1 tag curve_1`

(read x stress for shell 99 middle surface in-plane integration point 1)

`shell 99 stress xx surface middle ipoint 1 tag curve_1`

(read x stress for shell 99 middle surface in-plane integration point 1)

`shell 99 stress xx surface 5 ipoint 2 tag curve_1`

(read x stress for shell 99 layer 5 in-plane integration point 2)

### 7.4.4.3 Specifying integration points for Solids

In recent versions of LS-DYNA it is possible to write out data at all 8 integration points or fully integrated solid elements.

By default T/HIS will automatically read the average value for each element. If the element isn't fully integrated then the data for the 1st point will be used, if it is fully integrated and has multiple integration points then the average value will be calculated.

extra word #1	extra word #2	notes
ipoint	n	Specifies the solid integration point to read data from.  If this option isn't specified then the centre value will be selected. If the element is fully integrated then the average value will be calculated from all 8 values

e.g. **solid 99 stress xx tag curve\_1**

(read x stress for solid 99 centre value)

**solid 99 stress xx ipoint 1 tag curve\_1**

(read x stress for solid 99 integration point 1)

#### 7.4.4.4 Selecting data at element nodal positions

In recent versions of LS-DYNA it is possible to write out data for Solid, Shells and Thick Shells that has been extrapolated from the integration points to the elements nodes

For Shells the values at all through thickness layers can be extrapolated to the nodes. For Thick Shells the bottom surface values are extrapolated to nodes 1-4 and the top surface values are extrapolated to nodes 5-8.

extra word #1	extra word #2	notes
node	n	Specifies the element node number to read data for

e.g. **shell 99 stress xx node 3**

(read x stress for shell 99 middle surface extrapolated to node 3)

**shell 99 stress xx surface 5 node 1 tag curve\_1**

(read x stress for shell 99 layer 5 extrapolated to node 1)

**tshell 99 stress xx node 7 tag curve\_1**

(read x stress for thick shell 99 top surface extrapolated to node 7)

**solid 99 stress xx node 4 tag curve\_1**

(read x stress for solid 99 extrapolated to node4)

## 7.4.5 Data extraction reference table

Data type	Keyword	Second word	Third word	Fourth word	Description
Airbag	Air	Airbag id	[pr]essure	-	pressure
			[vo]lume	-	volume
			[ie]	-	internal energy
			[in]	-	mass flow rate in
			[ou]	-	mass flow rate out
			[tm]	-	total mass
			[de]nsity	-	Density
			[sa]	-	Surface area
			[te]mp	-	Gas temperature
Airbag Part Data (ABSTAT_CPM)	ab_part	Airbag id	Part id	[rf]	Reaction force
				[pr]essure	pressure
				maf	Mass flow rate through fabric
				mav	Mass flow rate through vent
				ta	Total area
				[un]blocked	Unblocked area
Airbag Sensors (CPM_SENSOR)	ab_sensor	Sensor id		[te]mperature	Temperature
				xc	X coord
				yc	Y coord
				zc	Z coord
				vx	X Velocity
				vy	Y Velocity
				vz	Z Velocity
				vm	Velocity Magnitude
				[pr]essure	pressure
				[de]nsity	Density
				[te]mp	Gas temperature

Beam	Bea	Beam id	[n]ormal	x	Axial force
				y	Shear force in Y
				z	Shear force in Z
			[m]oment	y	Moment in Y
				z	Moment in Z
				x	Torsional moment
			[stra]in	-	Axial strain
			[e]nergy	p1	Bending energy: end 1
				p2	Bending energy: end 2
			[r]otation	y1	Y rotation: end 1
				y2	Y rotation: end 2
				z1	Z rotation: end 1
				z2	Z rotation: end 2
				x	Torsional rotation
			[b]ending	y1	Y Bending moment: end 1
				y2	Y Bending moment: end 2
				z1	Z Bending moment: end 1
				z2	Z Bending moment: end 2
			[e]nergy	a	Axial collapse energy
				i	Internal energy
			[stre]ss	x	Axial stress
				xy	XY Shear stress
				zx	ZX Shear stress
			[eff]	-	Effective plastic strain
			[exx]	-	Axial strain
			[e]xtra	##	Extra data ##
			[di]screte	dx	Axial displacement
				dy	Displacement in Y
				dz	Displacement in Z
				rx	Axial rotation
				ry	Rotation in Y
				rz	Rotation in Z"
Boundary	Bou	Boundary id	[n]odal loads	fx	Applied X Force
				fy	Applied Y Force
				fz	Applied Z Force
				fm	Applied Resultant force
				e	Energy from applied force
			[ri]gid body loads	fx	Applied X Force
				fy	Applied Y Force
				fz	Applied Z Force
				fm	Applied Resultant force
				e	Energy from applied force
			[p]ressure nodal loads	fx	Applied X Force
				fy	Applied Y Force
				fz	Applied Z Force
				fm	Applied Resultant force
				e	Energy from applied force
			[rv]elocity r-body loads	fx	BC motion X Force
				fy	BC motion Y Force
				fz	BC motion Z Force
				fm	Resultant BC motion force
				en	Energy from BC motion
				mx	BC motion X Moment
				my	BC motion Y Moment
				mz	BC motion Z Moment
				mm	BC Moment Magnitude
			[v]elocity nodal loads	fx	BC motion X Force
				fy	BC motion Y Force
				fz	BC motion Z Force
				fm	Resultant BC motion force
				e	Energy from BC motion

<b>Contact</b>	Con / Sli	Contact id	[f]orce	x	Master X force
				y	Master Y force
				z	Master Z force
				m	Master Force Magnitude
				xs	Slave X force
				ys	Slave Y force
				zs	Slave Z force
				ms	Slave Force Magnitude
			[e]nergy	t	Total energy (Slave + Master)
				s	Slave side energy
				m	Master side energy
				f	Frictional energy
			[g]eometric	fx	X force
				fy	Y force
				fz	Z force
				fm	Force Magnitude
				mx	Moment in X
				my	Moment in Y
				mz	Moment in Z
				mm	Moment Magnitude
<b>Cross section</b>	Cro / Sec	Section id	[f]orce	x	X force
				y	Y force
				z	Z force
				m	Force Magnitude
			[m]oment	x	Moment in X
				y	Moment in Y
				z	Moment in Z
				m	Moment Magnitude
			[c]entroid	x	X centroid coord
				y	Y centroid coord
				z	Z centroid coord
			[a]rea	-	Area of section
<b>FSI</b>	FSI	FSI id	[pr]essure	-	pressure
			[f]orce	x	X force
				y	Y force
				z	Z force
				m	Force Magnitude
			[po]rous	-	Porous Leakage
			[m]ass	-	Mass Flux



Joint	Joi	Joint id	[f]orce	x	X force
				y	Y force
				z	Z force
				m	Force Magnitude
			[m]oment	x	Moment in X
				y	Moment in Y
				z	Moment in Z
				m	Moment Magnitude
			<b>*CONSTRAINED_JOINT_STIFFNESS_GENERALIZED</b>		
			[ph]i	an	Phi angle
				dt	d(Phi)/dt
				st	Phi stiffness moment
				da	Phi damping moment
				to	Phi total moment
			[th]eta	an	Theta angle
				dt	d(Theta)/dt
				st	Theta stiffness moment
				da	Theta damping moment
				to	Theta total moment
			[ps]i	an	Psi angle
				dt	d(Psi)/dt
				st	Psi stiffness moment
				da	Psi damping moment
				to	Psi total moment
			[ge]neralized	en	Total joint energy
			<b>*CONSTRAINED_JOINT_STIFFNESS_FLEXION-TORSION</b>		
			[al]pha	an	Alpha angle
				dt	d(Alpha)/dt
				st	Alpha stiffness moment
				da	Alpha damping moment
				to	Alpha total moment
			[be]ta	an	Beta angle
				dt	d(Beta)/dt
				st	Beta stiffness moment
				da	Beta damping moment
				to	Beta total moment
			[ga]mma	an	Gamma angle
				dt	d(Gamma)/dt
				fa	Gamma scale factor
			[fl]exion	en	Total joint energy
Joint	Joi	Joint id	<b>*CONSTRAINED_JOINT_STIFFNESS_TRANSLATIONAL</b>		
			[tr]anslational	xd	X displacement
				dxdt	d(X)/dt
				yd	Y displacement
				dydt	d(Y)/dt
				zd	Z displacement
				dzdt	d(Z)/dt
				xsf	X stiffness
				xdf	X damping
				xtf	X total
				ysf	Y stiffness
				ydf	Y damping
				ytf	Y total
				zsf	Z stiffness
				zdf	Z damping
				ztf	Z total
			en		Total joint energy

Node	No	Node id	[te]mperature	x	Temperature
			[to]p	temperature	Top Surface Temperature
			[bo]ttom	temperature	Bottom Surface Temperature
			[d]isplacement	x	X Displacement
				y	Y Displacement
				z	Z Displacement
				m	Displacement Magnitude
			[v]elocity	x	X Velocity
				y	Y Velocity
				z	Z Velocity
				m	Velocity Magnitude
			[a]cceleration	x	X Acceleration
				y	Y Acceleration
				z	Z Acceleration
				m	Acceleration Magnitude
			[c]oord	x	Current X coord
				y	Current Y coord
				z	Current Z coord
				m	Current Vector
			[b]asic	x	Basic X coord
				y	Basic Y coord
				z	Basic Z coord
				m	Basic Vector
			[r]otation	x	X rotation
				y	Y rotation
				z	Z rotation
				m	Rotation Magnitude
				vx	X rotational velocity
				vy	Y rotational velocity
				vz	Z rotational velocity
				vm	Rotation Vel Magnitude
				ax	X rotational acceleration
				ay	Y rotational acceleration
				az	Z rotational acceleration
				am	Rotation Accel Magnitude
			force	x	X force
				y	Y force
				z	Z force
				m	Force Magnitude
			[e]nergy	-	Energy
Node Group	Ng	Group id	force	x	X force
				y	Y force
				z	Z force
				m	Force Magnitude
Part	Pa	Part id	[k]inetic e	-	Kinetic energy
			[i]nternal e	-	Internal energy
			[h]ourglass e	-	Hourglass energy
			[t]otal e	-	Total energy
			[mx]	-	X momentum
			[my]	-	Y momentum
			[mz]	-	Z momentum
			[x] velocity	-	Average X velocity
			[y] velocity	-	Average Y velocity
			[z] velocity	-	Average Z velocity
			[am]	-	Added mass
			[ek]	-	Eroded Kinetic energy
			[ei]	-	Eroded Internal energy
Part group	Gro	Group id	[k]inetic e	-	Kinetic energy
			[i]nternal e	-	Internal energy
			[h]ourglass e	-	Hourglass energy
			[t]otal e	-	Total energy
			[am]	-	Added mass

Pulleys	Pul	Pulley id	[f]orce	-	Force
			[s]lip	-	Slip
			[r]ate	-	Slip Rate
			[a]ngle	-	Wrap Angle
Retractor	Ret	Retractor id	[f]ore	-	Force
			[p]ullout	-	Pullout
			[f]op	-	Force v Pullout
Rigid wall	Rig / Wall	Wall id	[n]or mal force	-	Normal force
			[x] force	-	Global X force
			[y] force	-	Global Y force
			[z] force	-	Global Z force
			[e] energy	-	Energy
Rigid wall Segment	Rigid_seg	Wall id	Segment id	[x] force	Global X force
				[y] force	Global Y force
				[z] force	Global Z force
Rigid part / NRB	rpa / nrb	Part id	[d]is placement	x	X Displacement
				y	Y Displacement
				z	Z Displacement
				m	Displacement Magnitude
			[v]elocity	x	X Velocity
				y	Y Velocity
				z	Z Velocity
				m	Velocity Magnitude
			[a]cceleration	x	X Acceleration
				y	Y Acceleration
				z	Z Acceleration
				m	Acceleration Magnitude
			[c]ord	x	X chord
				y	Y chord
				z	Z chord
			[r]otation	x	X rotation
				y	Y rotation
				z	Z rotation
				m	Rotation Magnitude
				VX	X rotational velocity
				ay	Y rotational velocity
				viz	Z rotational velocity
				am	Rotation Vel Magnitude
				ax	X rotational acceleration
				ay	Y rotational acceleration
				Az	Z rotational acceleration
				am	Rotation Axle Magnitude

Rigid part / NRB	rpa / nrb	Part id	[dc]os	11	Direction Cosine 11
				12	Direction Cosine 12
				13	Direction Cosine 13
				21	Direction Cosine 21
				22	Direction Cosine 22
				23	Direction Cosine 23
				31	Direction Cosine 31
				32	Direction Cosine 32
				33	Direction Cosine 33
			[lad]isplacement (local)	x	Local X Displacement
				y	Local Y Displacement
				z	Local Z Displacement
			[lv]elocity (local)	x	Local X Velocity
				y	Local Y Velocity
				z	Local Z Velocity
			[la]cceleration (local)	x	Local X Acceleration
				y	Local Y Acceleration
				z	Local Z Acceleration
			[or]otation (local)	x	Local X rotation
				y	Local Y rotation
				z	Local Z rotation
				VX	Local X rotational vel
				ay	Local Y rotational vel
				viz	Local Z rotational vel
				ax	Local X rotational axle
				ay	Local Y rotational axle
				Az	Local Z rotational axle
Seat belt	Sea / Bel	Belt id	[fo]rce	-	Force
			[s]train	-	Strain
			[fvs]	-	Force v Strain
			[l]ength	-	Current Length
Shell	Sh	Shell id	[stre]ss	xx	Stress in XX
				yy	Stress in YY
				zz	Stress in ZZ
				xy	Stress in XY
				yz	Stress in YZ
				zx	Stress in ZX
				mx	MAX principal stress
				mn	MIN principal stress
				ms	MAX shear stress
				vm	von Mises stress
				av	Average stress (Pressure)
			[stra]in	xx	Strain in XX
				yy	Strain in YY
				zz	Strain in ZZ
				xy	Strain in XY
				yz	Strain in YZ
				zx	Strain in ZX
				ma	MAX principal strain
				mi	MIN principal strain
				sh	MAX shear strain
				vm	von Mises strain
				av	Average strain
			[pla]stic	ef	Effective plastic strain
			[m]oment	x	Moment in X
				y	Moment in Y
				xy	Moment in XY
			[f]orce	sx	Shear force in X
				sy	Shear force in Y
				nx	Normal force in X
				ny	Normal force in Y
				nxy	Normal force in XY
			[t]hickness	-	Thickness
			[i]nternal	-	Internal energy density
			[e]xtra	##	Extra data ##

<b>Slipring</b>	Slp	Slipring id	[p]ullout	-	Pull through
<b>Solid</b>	yyp	Solid id	[stre]ss	xx	Stress in XX
				yy	Stress in YY
				zz	Stress in ZZ
				xy	Stress in XY
				yz	Stress in YZ
				zx	Stress in ZX
				mx	MAX principal stress
				mn	MIN principal stress
				ms	MAX shear stress
				vm	von Mises stress
				av	Average stress (Pressure)
			[stra]in	xx	Strain in XX
				yy	Strain in YY
				zz	Strain in ZZ
				xy	Strain in XY
				yz	Strain in YZ
				zx	Strain in ZX
				ma	MAX principal strain
				mi	MIN principal strain
				sh	MAX shear strain
				vm	von Mises strain
				av	Average strain
			[pla]stic	ef	Effective plastic strain
			[e]xtra	##	Extra data ##
<b>SPC</b>	SPC	SPC id	[f]orce	x	X force
				y	Y force
				z	Z force
				m	Force Magnitude
			[m]oment	x	Moment in X
				y	Moment in Y
				z	Moment in Z
				m	Moment Magnitude
<b>SPH</b>	SPH	SPH id	[d]ensity	-	Density
			[stra]in	xx	Strain in XX
				yy	Strain in YY
				zz	Strain in ZZ
				xy	Strain in XY
				yz	Strain in YZ
				zx	Strain in ZX
			[stre]ss	ef	Effective Stress
				xx	Stress in XX
				yy	Stress in YY
				zz	Stress in ZZ
				xy	Stress in XY
				yz	Stress in YZ
				zx	Stress in ZX
			[l]ength	-	Smoothing Length

Spotweld	Sw	Spotweld id	[co]nstrained	[a]xial	Axial force
				[s]hear	Shear force
				[l]ength	Length
				[f]ailure	Failure (failed if > 1.0)
				[ma]ximum	Maximum failure value
				[t]ime	Failure Time
			[ge]neralised	[a]xial	Axial force
				[s]hear	Shear force
				[l]ength	Length
				[f]ailure	Failure (failed if > 1.0)
				[ma]ximum	Maximum failure value
				[t]ime	Failure Time
			[sp]otweld	[a]xial	Axial force
				[s]hear	Shear force
				[l]ength	Length
				[f]ailure	Failure (failed if > 1.0)
				[ma]ximum	Maximum failure value
				[t]ime	Failure Time
				[m]oment	Resultant Moment
			[so]lid	[a]xial	Axial force
				[s]hear	Shear force
				[l]ength	Length
				[f]ailure	Failure (failed if > 1.0)
				[ma]ximum	Maximum failure value
				[t]ime	Failure Time
				ff	DC Failure Function
				nf	Normal Failure Term
				sf	Shear Failure Trem
				bf	Bending Failure Term
				[ar]ea	Spotweld Area
			[no]n-local	[a]xial	Axial force
				[s]hear	Shear force
				[l]ength	Length
				[f]ailure	Failure (failed if > 1.0)
				[ma]ximum	Maximum failure value
				[t]ime	Failure Time
			[ass]embly	[a]xial	Axial force
				[s]hear	Shear force
				[l]ength	Length
				[f]ailure	Failure (failed if > 1.0)
				[m]oment	Resultant Moment
				[t]ime	Failure Time
				ff	DC Failure Function
				nf	Normal Failure Term
				sf	Shear Failure Trem
				bf	Bending Failure Term
[ar]ea	Spotweld Area				
Spring	Sp / Da	Spring id		[f]orce	-
			[e]longation	-	Elongation
			[fve]	-	Force v Elongation
			[en]ergy	-	Energy
			[m]oment	-	Moment
			[r]otation	-	Rotation
			[mvr]	-	Moment v Rotation
			[x] force	-	Global X force
			[y] force	-	Global Y force
			[z] force	-	Global Z force
			[mx]	-	Moment in X
			[my]	-	Moment in Y
			[mz]	-	Moment in Z
			[re]nergy	-	Rotational Energy

Subsystem	Ss	Subsystem id	[k]inetic e	-	Kinetic energy
			[i]nternal e	-	Internal energy
			[h]ourglass e	-	Hourglass energy
			[kr]	-	Kinetic Energy Ratio
			[ir]	-	Internal Energy Ratio
			[mx]	-	X momentum
			[my]	-	Y momentum
			[mz]	-	Z momentum
Thick Shell	Thi / Tsh	Tshell id	[stre]ss	xx	Stress in XX
				yy	Stress in YY
				zz	Stress in ZZ
				xy	Stress in XY
				yz	Stress in YZ
				zx	Stress in ZX
				mx	MAX principal stress
				mn	MIN principal stress
				ms	MAX shear stress
				vm	von Mises stress
				av	Average stress (Pressure)
			[stra]in	xx	Strain in XX
				yy	Strain in YY
				zz	Strain in ZZ
				xy	Strain in XY
				yz	Strain in YZ
				zx	Strain in ZX
				ma	MAX principal strain
				mi	MIN principal strain
				sh	MAX shear strain
				vm	von Mises strain
				av	Average strain
			[pla]stic	ef	Effective plastic strain
			[e]xtra	##	Extra data ##
Whole model	Wh	-	[dt]	-	Time step
			[k]inetic e	-	Kinetic energy
			[i]nternal e	-	Internal energy
			[sw]	-	Stonewall energy
			[j]oint e	-	Joint internal energy
			[sp]ring e	-	Spring and damper energy
			[h]ourglass e	-	Hourglass energy
			[sy]stem e	-	System damping energy
			[si]	-	Sliding interface energy
			[ew]	-	External work
			[rb]	-	Rigid Body stopper energy
			[t]otal e	-	Total energy
			[er]	-	Total/initial energy
			[x] velocity	-	Average X velocity
			[y] velocity	-	Average Y velocity
			[z] velocity	-	Average Z velocity
			[cy]cle time	-	Time per zone cycle
			[am]	---	Added mass
			[pm]	-	%age Mass increase
			[ek]	-	Eroded Kinetic energy
			[ei]	-	Eroded Internal energy
			[eh]	-	Eroded Hourglass energy
			[ewoe]	-	Energy Ratio w/o Eroded
			[m]ass	-	Mass

TRACERS	Tr	Tracer ID	[d]isplacement	x	Current X coord
				y	Current Y coord
				z	Current Z coord
				m	Current Vector
			[v]elocity	x	X Velocity
				y	Y Velocity
				z	Z Velocity
				m	Velocity Magnitude
			[stre]ss	xx	Stress in XX
				yy	Stress in YY
				zz	Stress in ZZ
				xy	Stress in XY
				yz	Stress in YZ
				zx	Stress in ZX
			EFP	-	
			(de)nsity	-	Density
			rvol	-	Relative Volume
			ac[tive]	-	Active

### 7.4.5.1 Defining Surfaces / Integration points for data extraction

Some data components can be written out at multiple locations

In recent versions of LS-DYNA it is possible for each element to write out multiple values for some data components.

For fully integrated Shells and Thick Shells values can be written out for all 4 in-plane integration points in each through thickness location. In addition to the integration point values it is also possible to write out data that has been extrapolated from the integration points out to the shells nodes.

For fully integrated solid elements data can also be written out for all 8 integration points and values can also be extrapolated to the elements nodes.

To select these additional values the entity ID's specified in a FAST-TCF scripts can be modified as follows.

Solids	n	Average value for solid (default)
	n@X	Value at integration point X ( 0 < X < 8)
	n@-X	Value at node X ( 0 < X < 8)
Shells	n	Average value for shell (default)
	n@X	Value at integration point X ( 0 < X < 4)
	n@-X	Value at node X ( 0 < X < 4)
Shells	n	Average value for thick shell (default)
	n@X	Value at integration point X ( 0 < X < 4)
	n@-X	Value at node X ( 0 < X < 8)

e.g. **solid 10**

(solid 10 - average value)

**solid 20@5**

(solid 20 - data from 5th integration point)

**shell 20@-3**

(shell 20 - data extrapolated to shells 3rd node)



## 7.5 UNITS

From version 9.4 onwards T/HIS can automatically add unit information to graph labels and it can convert results from one unit system to another.

Each model in T/HIS can have a Unit System defined for it and a separate Unit System can be defined for displaying results. T/HIS will automatically convert results from the model Unit System to the display Unit System. T/HIS has 6 built in unit systems

Unit System name	Units
U1	m,kg,s
U2	mm,Tonnes,s
U3	mm,kg,ms
U4	mm,gm,ms
U5	ft,slug,s
U6	m,Tonnes,s

### 7.5.1 Setting the unit system for a model

To set the unit system for a model

Keyword	second word	third word	fourth word	notes
unit	model	n	U1	Set the unit system for model 'n' to U1
			U2	Set the unit system for model 'n' to U2
			U3	Set the unit system for model 'n' to U3
			U4	Set the unit system for model 'n' to U4
			U5	Set the unit system for model 'n' to U5
			U6	Set the unit system for model 'n' to U6
	all		as above	Set the unit system for all models

### 7.5.2 Setting the DISPLAY unit system

To set the display unit system

Keyword	second word	third word	notes
unit	display	U1	Set the display unit system to U1
		U2	Set the display unit system to U2
		U3	Set the display unit system to U3
		U4	Set the display unit system to U4
		U5	Set the display unit system to U5
		U6	Set the display unit system to U6

### 7.5.3 Curve Axis units

By default T/HIS will automatically set the Unit System for any curves read from a model to those of the model. In addition to setting the curve Unit System T/HIS will automatically set a unit type for the X and Y axis of the curve. These unit types are maintained through curve operations so that the correct units can be displayed for each curve.

The X and Y Axis units for a curve can be manually set if required.

Keyword	second word	third word	additional words	notes
unit	cx	curve #1	curve #2 to curve #n	Unit ## name ## ends the curve list Set the X axis unit for curves
		*	##	Unit ## name ## ends the curve list Set the X axis unit for all curves
	cy	curve #1	curve #2 to curve #n	Unit ## name ## ends the curve list Set the Y axis unit for curves
		*	##	Unit ## name ## ends the curve list Set the Y axis unit for all curves

The Unit name can be any of the following

Time	Rotation	Momentum	Energy Den
Energy	Rot Vel	Density	Mass Flow
Work	Rot Accel	Stress	Frequency
Temperature	Length	Strain	Power
Displacement	Area	Force	Thermal Flux
Velocity	Volume	Moment	Force width
Accel	Mass	Pressure	Moment width

## 7.5.4 Curve Unit Systems

If a curve has been read in from any source other than a model then the Unit System can also be set.

Keyword	second word	third word	additional words	notes
unit	cu	curve #1	curve #2 to curve #n	Unit System name ## ## ends the curve list Set the Unit System for curves
		*	##	Unit System name ## ## ends the curve list Set the Unit System for all curves

## 7.5.5 Other UNIT options

If a CSV file is written out from within a FAST-TCF script ([see section 7.14](#)) then by default it will contain rows containing UNIT information for the curves if UNITS have been defined.

Some third party applications and scripts can not read T/HIS CSV files containing this additional UNIT information correctly. The following option can be added to FAST-TCF scripts to turn on and off the output of this additional information.

Keyword	second word	third word	notes
unit	csv	on	Turns on the output of UNIT information to CSV files
		off	Turns off the output of UNIT information to CSV files

## 7.6 CURVE TAGS

In FAST-TCF any operation that uses one or more curves as an input can reference the curve using either the curve number or a curve tag. **The use of curve Tags is strongly recommended as it enables scripts to be easily modified and sections added / deleted without having to renumber all the curve references within the script.**

Curve tags are defined for a curve by adding the keyword TAG to the data extraction command followed by the tag.

<b>e.g.</b>	<b>node</b>	<b>42</b>	<b>force y_dir</b>	<b>tag curve_1</b>
	(node)	(i.d. 42)	(force in y-direction)	(tag the curve as "curve_1")
	<b>node</b>	<b>"end of roof"</b>	<b>accel z</b>	<b>tag point_2</b>
	(node)	(i.d. "end of roof")	(z acceleration)	(tag the curve as "point_2")

Tags cannot begin with a numeric character, e.g. tag 1\_curve is not allowed.

If a tag is not specified for a curve then FAST-TCF will automatically generate a tag for the curve using the T/HIS curve number as the TAG.

The TAG for a curve can be redefined at anytime within a script using the "tag" command ([see section 7.10.1](#)) for more details. Once a curve tag has been redefined the original definition should not be used in any following commands - a curve can only have 1 TAG defined at any time.

### 7.6.1 Tagging curves from a T/HIS curve file

Curves read in from a T/His curve file can be tagged by referring to each curve in the file using a negative number:

<b>e.g.</b>	<b>tag</b>	<b>-1</b>	<b>curve_1</b>
		(1st curve in the curve file)	(tag as "curve_1")
	<b>tag</b>	<b>-2</b>	<b>curve_2</b>
		(2nd curve in the curve file)	(tag as "curve_2")

If curves are read in from a T/HIS curve file then the FAST-TCF tag will be generated using the following rules.

1. If the data extraction command contains a TAG option then that TAG will be used (as above).
2. If the curve file contains curve tags then they will be used if the data extraction command DOES NOT contain a TAG option.
3. If no tags are specified in the file or in the data extraction command then T/His will automatically tag each curve as '#', where # is the internal T/HIS curve number.

In the third case, if for example there are three curves already in T/His, the curves read in from the curve file will be tagged as '4', '5', '6', '7', etc. This limits how you can refer to these curves since would not be able to multiply two curves together. For example the command '**op mul 4 5 tag new\_curve**' would multiply the curve tagged as '4' by the number 5, not by the curve tagged as '5'.

To avoid this limitation you will need to tag your curves using either the syntax explained above or by specifying a tag in the curve file.

### 7.6.2 Tagging multiple curve outputs

From version 9.2 onwards multiple curve outputs can be generated from one FAST-TCF input line. Curve tags and labels can be specified for multiple curves using the following special syntax (note this only works on multiple curves):

- If the user specifies a wildcard in the tag or label (a "\*"), then FAST-TCF will substitute the wildcard for the number of the curve outputted (starting from 1).
- If the user specifies a "##" then the entity ID is substituted in its place which is useful if the user knows what entities are expected on output.

<b>e.g.</b>	<b>node 5:last</b>	<b>accel mag</b>	<b>tag node_*</b>	<b>lab Head Accn *</b>
	(node IDs. 5 to last)	(accel mag)	tags = node_1, node_2, etc	labels = Head Accn 1, Head Accn 2, etc
	<b>node 10:20</b>	<b>accel mag</b>	<b>tag node_##</b>	<b>lab Head Accn ##</b>
	(nodes 10 to 20)	(accel mag)	tags = node_10, node_11, etc	labels = Head Accn 10, Head Accn 11, etc

## 7.6.3 Using Wildcards

A number of T/HIS functions and operations can be applied to multiple curves in a single command by specifying multiple curve tags using wildcards.

From version 10.0 onwards the following wildcards are supported

Wildcard	Matches
*	1 or more characters
?	a single character
[a-e]	matches a single character against a range of characters , 'a','b','c','d' or 'e'
[abc]	matches a single character against a list of characters, 'a', 'b' or 'c'

**e.g. operate multiple x\_disp\_\* 10 tag x\_mul\_\***

(Multiple all curves with a tag starting with "x\_disp\_" by 10 and tag the outputs as x\_mul\_1, x\_mul\_2 ... - see [Section 7.8](#) for more details)

**display x\_disp\_\***

(Display all curves with a tag starting with "x\_disp\_" - see [Section 7.12.4](#) for more details)

**copy curve\_file.cur x\_disp\_\***

(Write all curves with a tag starting with "x\_disp\_" to a file called "curve\_file.cur"- see [Section 7.14](#) for more details)

**csv curve\_file.csv curve\_1? curve\_3[0-3]**

Write curves with tags curve\_10, curve\_11, curve\_12 .... and curves with tags curve\_30, curve\_31, curve\_32, curve\_33 to a CSV file called "curve\_file.csv"- see [Section 7.14](#) for more details)

## 7.6.4 Using Curve Numbers

Although it is not recommended curves can be referenced using the internal curve number instead of the curve tag. If for example the 1st curve generated by a script has the tag "curve\_1" then the following 2 commands are identical.

**e.g. operate multiple curve\_1 10 tag x\_mul\_\***  
**operate multiple #1 10 tag x\_mul\_\***

If curve numbers are used within a script then T/HIS will automatically offset the curve numbers in the script by the number of curves T/HIS already has defined before the script is executed.

**e.g. operate multiple #1 10 tag x\_mul\_\***

would multiply internal curve number 1 by 10 if T/HIS didn't contain any curve definitions when the script was run.

If T/HIS already contained 100 curves then the same command would multiply internal curve 101 by 10.

This means it is possible to play a script containing curve numbers multiple times within a session without having to either delete all the existing curves or modify the script each time.

## 7.7 CURVE GROUPS

Curve groups can be defined within FAST-TCF scripts using the **cgroup** keyword. After a curve group has been defined in a FAST-TCF script it can then be used as an input to some FAST-TCF commands. Each curve group should be given a unique name within the FAST-TCF script.

Keyword	Second word	Third word	following word	notes
cgroup	create	name	-	Create a curve group called "name". If the name contains any spaces then it should be enclosed in quotes ("name with space")
	add	name	curve list	Adds a list of curves to the curve group called "name". If the name contains any spaces then it should be enclosed in quotes ("name with space"). The curve list should be a list of curve tags.
	remove	name	curve list	Removes a list of curves from the curve group called "name". If the name contains any spaces then it should be enclosed in quotes ("name with space"). The curve list should be a list of curve tags.

e.g. **cgroup create group\_1**  
 (Create a curve group called "group\_1")

**cgroup add group\_1 curve\_1 curve\_2**  
 (Add curves with tags "curve\_1" and "curve\_2" to group "group\_1")

**cgroup create "Group 2"**  
 (Create a curve group called "Group 2")

**cgroup add "Group 2" curve\_1\***  
 Add all curves with a curve tag containing "curve\_1" to group "Group 2"

**cgroup remove "Group 2" curve\_11**  
 Remove curve with tag "curve\_11" from group "Group 2"

To use a curve group as the input to another FAST-TCF command the curve group name is preceded by an &. If a curve group name contains spaces then the name should be enclosed in double quotes and the & should be before the first ".

e.g. **operate multiple &group\_1 10 tag output\_\***  
 (Multiple all curves in curve group "group\_1" by 10 and tag the outputs as output\_1, output\_2 ...)

**operate multiple &"Group 2" 10 tag output\_\***  
 (Multiple all curves in curve group "Group 2" by 10 and tag the outputs as output\_1, output\_2 ...)

Curve Groups can currently be used as

- The first curve input in all of the [operate](#) commands
- Within the list of curves specified as input to [curve range](#) functions.
- To select a group of curves for the [display](#) command.
- Outputting curves to T/HIS curve files and CSV files.

## 7.8 PERFORMING FAST-TCF CURVE OPERATIONS

Description	keyword	following words
Curve operation	oper	oper command + necessary words (depending on operation)

Many curve processing operations and functions are available. The syntax is common for all types of curve operation:

1. the first word is 'oper' and is followed by:
2. the operation/function name e.g. ADD, int.
3. the required number of arguments for the operation, e.g. ADD requires two arguments, a curve and either a curve or a value.
4. the remainder of the line may contain optional requests.
5. any optional requests can occur after the arguments.
6. curve numbers must be in the format: #<curve number>
7. An output curve is always needed - for operation commands such as hic, hicc, tti, 3ms, err, the curve will be copied and the operation is executed on the copied curve.
8. A curve tag containing a wildcard or a curve group can be specified as the first curve input for any curve operation. If a curve tag contains a wildcard or if a curve group is specified then the curve operation will be repeated for each curve that either the tag matches or is in the curve group.

e.g.    **oper hic    node\_acc 1.0            15E-3            label Hic-ed node accn**  
           (hic)            (curve tag)    (scale=1.0)    (15ms period)    (label)

In T/HIS 9.2 onwards, the user can operate on multiple input curves (only the first curve can be multiple at the moment) using the wildcard "\*". For example, to multiply all curves starting with the tag **acc**:

e.g.    **oper mul    acc\*            9810.0**  
           (multiply)    (on all curves with tag acc\*)

### 7.8.1 Standard operation commands

Description	keyword	operation command	following word #1	following word #2	additional words	notes
Absolute value	oper	abs	curve #1	-	-	
Add Y	oper	add	curve #1	curve #2 or constant	-	
Add X	oper	adx	curve #1	curve #2 or constant	-	
Clip curve	oper	cli	curve #1	x min value / "auto"	x max value / "auto" y min value / "auto" y max value / "auto"	Input requires all 4 values, "auto" sets the value automatically
Combine	oper	com	curve #1	curve #2	-	
Concatenate	oper	cat	curve #1	curve #2	-	
Derivative	oper	dif	curve #1	-	-	
db	oper	db	curve #1	reference value		Convert a curve to dB
db(A)	oper	dba	curve #1	narrow		Apply narrow band A weighting
				octave		Apply octave band A weighting
Div Y	oper	div	curve #1	curve #2 or constant	-	
Div X	oper	dix	curve #1	curve #2 or constant	-	
Error calculation	oper	err	curve #1	curve #2	-	Value is stored with the output curve
Integral	oper	int	curve #1	-	-	
Least squares	oper	lsq	curve #1	-	-	
Map	oper	map	curve #1	curve #2	-	
Mul Y	oper	mul	curve #1	curve #2 or constant	-	

Mul X	oper	mux	curve #1	curve #2 or constant	-		
Normalise	oper	nor	curve #1	-	-		
Octave	oper	oct	curve #1	octave	rms	linear	Convert a curve from "narrow" band to either Octave or 1/3rd Octave bands.  Value for each band can be calculated using either mean or RMS values, and the input can either be linear or in dB.
						db	
				third	mean	linear	
						db	
					rms	linear	
	db						
Order	oper	ord	curve #1	-	-		
Reciprocal	oper	rec	curve #1	-	-		
Reverse	oper	rev	curve #1	-	-		
Rolling average	oper	r-av	curve #1	averaging window	-		If the averaging window is undefined or set to 0.0 then the y-values at each point are calculated by averaging all of the proceeding curve points.  If the averaging window is set to T then the y-values at each point are calculated by averaging between -T/2 and +T/2.
Smooth	oper	smo	curve #1	smoothing factor	-		Factor must be an integer
Stress	oper	str	curve #1	"true" or "engineering"	-		
Sub Y	oper	sub	curve #1	curve #2 or constant	-		
Sub X	oper	sux	curve #1	curve #2 or constant	-		
Translate	oper	tra	curve #1	X value	Y value		
Vector 2D	oper	v2d	curve #1	curve #2	-		
Vector mag	oper	vec	curve #1	curve #2	curve #3		
Window	oper	win	curve #1	"han", "cos", "exp"	lead in (only for "cos" option)	Writes out 2 curves	
Zero curve (X and Y)	oper	zer	curve #1	-	-		Shifts curve to 0,0 (X and Y values)
Zero curve (X only)	oper	zero_x	curve #1	-	-		Shift curve to 0,Y (X only)
Zero curve (Y only)	oper	zero_y	curve #1	-	-		Shift curve to X,0 (Y only)

## 7.8.2 Maths operation commands

Description	keyword	operation command	following word #1	following word #2	additional words	notes
<b>Arc cosine</b>	oper	acos	curve #1	-	-	
<b>Arc sine</b>	oper	asin	curve #1	-	-	
<b>Arc tangent</b>	oper	atan	curve #1	-	-	
<b>Cosine</b>	oper	cos	curve #1	-	-	
<b>Log base 10</b>	oper	log10	curve #1	-	-	
<b>Log base 10 (X)</b>	oper	log10x	curve #1	-	-	
<b>Natural Exp</b>	oper	exp	curve #1	-	-	
<b>Natural log</b>	oper	log	curve #1	-	-	
<b>Natural log (X)</b>	oper	logx	curve #1	-	-	
<b>Power</b>	oper	pow	curve #1	nth power	-	
<b>Sine</b>	oper	sin	curve #1	-	-	
<b>Square root</b>	oper	sqr	curve #1	-	-	
<b>Tangent</b>	oper	tan	curve #1	-	-	

### 7.8.3 Automotive operation commands

Description	keyword	operation command	following word #1	following word #2	additional words	notes
Delta V	oper	acu	curve #1	offset	time period	
Acceleration severity index	oper	asi	Accn x curve #	Accn y curve #	Accn z curve #	word6 = acceleration conversion factor word7 = x limit word8 = y limit word9 = z limit
Butterworth filter	oper	but	curve #1	cut off frequency	order	
C60 filter	oper	c60	curve #1	-	-	
C180 filter	oper	c180	curve #1	-	-	
C600 filter	oper	c600	curve #1	-	-	
C1000 filter	oper	c1000	curve #1	-	-	
Clip value	oper	cva	curve #1	time window	Label displayed on screen ( <i>optional</i> )	Value is stored with the output curve
Exceedence	oper	exc	curve #1	auto / pos / neg	-	
Fir filter	oper	fir	curve #1	-	-	
Hic	oper	hic	curve #1	division scale factor	time period	Value is stored with the output curve
Hicd	oper	hicd	curve #1	division scale factor	time period	Value is stored with the output curve
Neck injury criteria	oper	nij	Shear curve #	Axial curve #	Moment curve #	word6 = Fzc tension word7 = Fzc compression word8 = Myc flexion word9 = Myc extension word10 = Distance from joint
Regularise	oper	reg	curve #1	new dt value	-	
THIV	oper	thi	Accn x curve #	Accn y curve #	Yaw rate curve #	word6 = Horizontal distance word7 = Lateral distance word8 = Head to vehicle distance
TTI	oper	tti	Upper rib curve #	Lower rib curve #	Lower spine curve #	Value is stored with the output curve
Viscous criteria ECER95	oper	vc	curve #1	constant A	constant B	ECER95 method
Viscous criteria IIHS	oper	vc2	curve #1	constant A	constant B	IIHS method
Curve Correlation (strict)	oper	corr	strict	curve #1	curve #2	Value is stored with the output curves
Curve Correlation (loose)	oper	corr	loose	curve #1	curve #2	Value is stored with the output curves
Weighted Integrated Factor Curve Correlation (WIFAC)	oper	wif	curve #1	curve #2	-	Value is stored with the output curve

### 7.8.4 Seismic operation commands

Description	keyword	operation command	following word #1	following word #2	additional words	notes
Accn to disp spectra	oper	ad	curve #1	-	-	
Accn to vel spectra	oper	av	curve #1	-	-	
Disp to vel spectra	oper	dv	curve #1	-	-	
Disp to accn spectra	oper	da	curve #1	-	-	
Vel to disp spectra	oper	vd	curve #1	-	-	



<b>Vel to accn spectra</b>	oper	va	curve #1	-	-	
<b>Baseline correction</b>	oper	bic	curve #1	-	-	
<b>Design spectrum</b>	oper	ds	curve #1	broadening factor	-	
<b>FFT</b>	oper	fft	curve #1	-	-	
<b>Non cumulative P.R.</b>	oper	ncp	curve #1	curve #2	-	
<b>Response spectrum</b>	oper	rs	curve #1	damping factor	sampling factor	Sampling must be either 30 or 70

## 7.8.5 Range of curve operation commands

<b>Description</b>	<b>keyword</b>	<b>operation command</b>	<b>following word #1</b>	<b>following word #2</b>	<b>additional words</b>	<b>notes</b>
<b>Average</b>	oper	ave	curve #1	curve #2 to curve #n	##	"##" ends the curve list
<b>Envelope</b>	oper	env	curve #1	curve #2 to curve #n	##	"##" ends the curve list
<b>Minimum</b>	oper	min	curve #1	curve #2 to curve #n	##	"##" ends the curve list
<b>Maximum</b>	oper	max	curve #1	curve #2 to curve #n	##	"##" ends the curve list
<b>Sum</b>	oper	sum	curve #1	curve #2 to curve #n	##	"##" ends the curve list
<b>Sum</b>	oper	sum	curve #1	curve #2 to curve #n	##	"##" ends the curve list

## 7.9 APPLYING EXTRA OPTIONS TO DATA REQUESTS

Extra options can be used after a data component extraction, or a curve operation. After the basic request for a particular component and particular entity have been made, the following extra data on the line is recognised to manipulate the curve further. This includes options to label a curve, scale it, write it out and so on.

Each request is executed in the order on the line, **if the curve label is used, it must be the last input on the line.**

```
e.g.  no 54      accel mag xsc 1000 ysc 0.0001    hic          lab Head Accn
      (node i.d. 54) (accel mag)  (scale x and y)      (obtain hic value)    (curve label)

      no 1       accel mag filter c60            append output.cur
      (node i.d. 1) (accel mag)  (filter with C60)      (append the curve to a file)

      no 1       accel mag tag node_1_acc
      (node i.d. 1) (accel mag)  (tag the curve "node_1_acc" for ease of use later in the script)
```

Description	extra option word	following word #1	following word #2	notes
3ms clip	3ms	-	-	Curve is squared and then square rooted to remove -ve values Curve is truncated around 3ms values - only 3ms part is left
Append into file	app	filename	-	Appends into curve file, if it doesn't exist - create it
Combine	com	curve #2	-	Y-value curve #1 vs X-value curve #2
Copy into file	cop	filename	-	Copy will overwrite any previous instance of the file
Error function	err	curve #2	-	
HIC	hic, hic15, hicc	-	-	Curve is squared and then square rooted to remove -ve values, an identical curve is outputted
Filtering	fil	fir	-	
		c60	-	
		c180	-	
		c600	-	
		c1000	-	
X scale factor	xsc	scale factor	-	
Y scale factor	ysc	scale factor	-	
Label	lab	label word #1	label word #2 etc	Keyword and label must be at the end of the line
Reference tag	tag	tag word	-	Invalid words: "style", "xax", "yax", "title"
ASCII file request	ASC	-	-	
LSDA file request	LSD	-	-	
THF file request	THF	-	-	
XTF file request	XTF	-	-	

### 7.9.1 Using extra options on multiple curve outputs

From version 9.2 onwards multiple curve outputs can be generated from one FAST-TCF input line. Unfortunately most of the extra options displayed below will NOT work on these multiple outputs. However, support has been added to allow tagging and labeling of all the multiple curves outputted in one go ([see section 7.6.1](#))

## 7.10 Setting properties for curves

The following options can be used to set up properties for curves.

### 7.10.1 Setting curve Labels, Titles and tags

Description	keyword	second word	third word	fourth word	notes
Curve Label	lab	curve # or tag	label word 1	label word 2 etc	Specifies a new curve label
Curve Tag	tag	curve # or tag	tag	-	Specifies a new curve tag
Curve Title	tit	curve # or tag	label word 1	label word 2 etc	Specifies a new curve title
Curve X axis label	xla	curve # or tag	label word 1	label word 2 etc	Specifies a new x -axis label
Curve Y axis label	yla	curve # or tag	label word 1	label word 2 etc	Specifies a new y-axis label
1st Y axis	y1	curve # or tag	-	-	puts the curve on the 1st y axis
2nd Y axis	y2	curve # or tag	-	-	puts the curve on the 2nd y axis

From version 9.4 onwards curve properties such as the minimum and maximum values can be displayed in the legend area as well as within the graph area.

The following commands use a new properties keyword and can be used to specify the font, colour and background used to display values as well as selecting which values are displayed on each curve.

Keyword	2nd word	3rd word	4th word	5th word	6th word	7th word	notes
properties	format	font	hm hb cm cb tm tb default	8 10 12 14 18 24 default	standard colour	-	sets up font used to display curve properties  fonts available: hm - helvetica medium cb - courier bold tm - times new roman medium etc...  font sizes in pt: 8, 10, 12 etc...
		background	standard colour	-	-	-	Set a background colour for the text
		transparency	integer (0-100)	-	-	-	Set the background transparency
		border	standard colour	on/off	-	-	Set a border colour round the text and turn it on/off
		arrow	on/off	-	-	-	Turn on/off a line connecting the text to the min/max value location
		num	y_only	-	-	-	Only display the y value
		num	x_y	-	-	-	Display both the x and y values on a single line
		num	xy	-	-	-	Display both the x and y values on separate lines
		value	<type>	-	-	-	Set the unit format to one of <i>automatic</i> , <i>general</i> , <i>scientific</i> for graph (n)
		precision	m	-	-	-	Set the number of decimal places displayed for the y axis values to (m) in graph (n)
properties	legend	format	off	-	-	-	Turn off the display of curve values in the legend area
			append	-	-	-	Append curve values (min,max,average ...) to the curve labels in the legend.
			2nd	-	-	-	Add a 2nd line to the legend for each curve containing the curve values (min,max,average ...).
		curve #1	curve #2	##	maximum	on/off	Turn on/off the display of one of the following curve properties in the legend. Input one or more curves and terminate the list with ##  maximum - display curve maximum value minimum - display curve minimum value average - display curve average value other - display other curve values
					minimum		
					average		
					other		

properties	curves	format	off	-	-	-	Turn off the display of curve values in the graph area
			summary	-	-	-	Display the minimum/maximum value for all of the curves currently visible
			all	-	-	-	Display minimum/maximum values for each curve that is currently visible
		summary	smaximum	on/of	-	-	Turns on/off the display of one of the following curve summary properties  smaximum - highlight the maximum value for all the curves displayed sminimum - highlight the minimum value for all the curves displayed lmaximum - label the maximum value for all the curves displayed lminimum - label the minimum value for all the curves displayed
			sminimum				
			lmaximum				
			lminimum				
		curve #1	curve #2	##	smaximum	on/off	Turns on/off the display of one of the following curve summary properties. Input one or more curves and terminate the list with ##  smaximum - highlight the maximum value for each curves sminimum - highlight the minimum value for each curves lmaximum - label the maximum value for each curves lminimum - label the minimum value for each curves other - label other curve values
					sminimum		
					lmaximum		
					lminimum		
					other		

## 7.11 Defining Datums

### 7.11.1 Creating Datum Definitions

The following options can be used to setup DATUM definitions

keyword	second word	notes
start datum		Starts a Datum definition
acronym	<i>acronym</i>	Specifies the datum acronym
label	<i>label</i>	Specifies the datum label
type	constant_x	Defines the datum as a constant x value
	constant_y	Defines the datum as a constant y value
	Points	Defined the datum as a set of x,y points
value	<i>value</i>	Specifies the value for a constant x or y datum
num_points	#points x1,y1 y2,y2	Specifies the number of points used to define a datum, followed by pairs of x,y values.
line_colour	colour (see section 7.12.1.2)	Specify the line colour used to display the datum line (or none)
line_style	style (see section 7.12.1.1)	Specifies the line style used to display the datum line (or none)
line_width	width (see section 7.12.1.3)	Specifies the line width used to display the datum line (or none)
fill_colour1	colour (see section 7.12.1.2)	Defines the colour used to fill above/right of the datum line
fill_colour2	colour (see section 7.12.1.2)	Defines the colour used to fill below/left of the datum line
label_font		Define the font used to display the label
label_size	8,10,12,14,18,24	Define the font point size used to display the label
label_colour	colour (see section 7.12.1.2)	Define the colour used to display the label
label_position	Above Centre	Position label at centre above line
	Above Left	Position label on left above line
	Above Right	Position label on right above line
	Below Centre	Position label at centre below line
	Below Left	Position label on left below line
	Below Right	Position label on right below line
	None	Turn off label display
	Middle Left	Position label on left in middle
	Top Left	Position label on left at top
	Bottom Left	Position label on left at bottom
	Middle Right	Position label on right in middle
	Top Right	Position label on right at top
	Bottom Right	Position label on right in middle
label_point	point number	Position label at datum point
end datum		Ends a Datum definition.

Each DATUM definition must start with a "start\_datum" keyword and end with a "end\_datum" keyword. Any lines between a "start\_datum" and "end\_datum" keyword that do not form part of a datum definition are ignored.

The following creates a DATUM definition at Y=1000.0 with a label "Hic Limit", the area below the line is filled in **GREEN** and the area above is filled in **RED**.

```

START_DATUM
ACRONYM datum_1
LABEL Hic Limit
TYPE constant_y
VALUE 1000.000000
LINE_COLOUR green
LINE_STYLE solid
LINE_WIDTH normal
FILL_COLOUR_1 red
FILL_COLOUR_2 green
LABEL_FONT default
LABEL_SIZE automatic
LABEL_COLOUR foreground
LABEL_POSITION default
END_DATUM

```

---

## 7.11.2 Adding Datum Lines to Graphs

Multiple DATUM definitions can be added to each graph using the datum acronym

Keyword	second word	notes
datum	acronym	Adds the datum with the acronym to the currently selected graphs

<code>datum maximum</code>	Add datum with the acronym "maximum" to all selected graphs
<code>layout graph select none</code>	Deselect all graphs ( <a href="#">see section 7.2</a> )
<code>layout graph select 1</code>	Select graph 1 ( <a href="#">see section 7.2</a> )
<code>datum maximum</code>	Add datum with the acronym "maximum" to graph 1 (currently selected)
<code>datum minimum</code>	Add datum with the acronym "minimum" to graph 1 (currently selected)

## 7.12 FAST-TCF IMAGE OUTPUT OPTIONS

The options to generate images can be split into 5 sections:

## Setting Curve Style

## Setting Curve Styles by Model

## Plot setup

## Curve Display

## Image Generation

### 7.12.1 Setting Curve Styles

Description	keyword	second word	following words
Plot style setup	style	style name	style options
Individual curve style	stylec	curve number or tag	style options

This section explains how to set up the styles for the curves in a plot. The two types of syntax available in the table above effect when and how the curves are styled.

The "plot style setup" (keyword style) allows the user to define a plot-specific styling that applies the styles to the curves only when they are requested for a plot. It is independent of the curve id, but dependent on the order the curves are requested in the plot command. The style is given its own "tag" which the user can request on the image FAST-TCF line.

This is useful for producing plots from FAST-TCF that all have the same curve appearance. For the following words, each space represents a new curve style definition. The styles for each curve are defined by the type keywords below, separated by commas.

e.g. style ENERGIES solid,green,norm dash,blue,heavy sol,bold,yel,500  
(style name) (curve #1) (curve #2) (curve #3)

When a plot is requested, FAST-TCF will apply the curve styles to the list of curves (in order) in the plot. So in the example above, the first curve would appear green, the second curve blue and the third yellow.

The "individual curve style" (keyword `stylec`) is the more traditional way of styling a curve that a T/HIS user would be more familiar with - FAST-TCF styles the single curve id instantly. The user can only define one style at a time.

e.g. `stylec #12`      `solid,green,norm`  
(style curve number 12) style to apply

### 7.12.1.1 Line Styles

Style options	word options	default
Line style	solid	solid
	dash	
	none	

### 7.12.1.2 Line Colours

Style options	word options	default
---------------	--------------	---------

<b>Line colour</b>	white	dependent on curve #
	red	
	green	
	blue	
	cyan	
	magenta	
	yellow	
	orange	
	turquoise	
	indigo	
	lime	
	sky	
	pink	
	black	
	foreground	
	background	

### 7.12.1.3 Line Width

Style options	word options	default
<b>Line width</b>	fine	normal
	normal	
	bold	
	heavy	

### 7.12.1.4 Line Symbols

Style options	word options	default
<b>Line symbols</b>	triangle	dependent on curve #
	square	
	diamond	
	hourglass	
	cross	
	circle	
	star	
	dot	
	null	
<b>Symbol frequency</b>	frequency number	-

## 7.12.2 Setting Curve Styles by Model

From version 11 onwards a new option can be used to colour all of the curves belonging to a model in a single operation.

Description	keyword	second word	following words
<b>Colour curves by model</b>	style_m	model number	style options

The available style options are exactly the same as for the `stylec` command ([see section 7.12.1](#))

**e.g. style\_m 2 solid,green,norm**

model number 2 style to apply

would set all the curves belonging to model 2 to solid, green lines using the default line thickness.

## 7.12.3 Plot setup

Description	keyword	following words
<b>Plot setup</b>	setup	plot setup words

These options set the appearance of any plots that are created afterwards. They are to do with the general appearance of



the plot rather than the curve itself. The curve appearances can be set up with the [style definition line](#) and on the [image plotting line](#). All following words must be on the same line. If the "on" or "off" is missed out from the following word (where applicable) then FAST-TCF will take the **opposite** to the default (this helps with backwards compatibility issues but can also make a script more compact).

e.g. **setup ax bold grid on line bold reverse**  
           (bold axes)      (grid on)      (bold lines)      (reverse foreground and background)

**setup double on border off show 3ms size 250**  
           (double axes on) (no border)      (3ms window on)      (size = 1000 x 650 pixels)

**setup fonts title hb 24 red**  
           (title: helvetica bold 24pt, in red)

Plot setup description	plot setup word	following word(s)	notes
Axis thickness	ax	fine normal bold heavy standard colour	for colours - see standard list below
Background	bac	standard colour	for colours - see standard list below
Border	bo	fine normal bold heavy standard colour on or off	for colours - see standard list below
Double yaxis	do	on or off	
Fix line styles	fix	on or off	this overwrites any style definitions
Fonts	fon	<div> <div> <div>[xl]label</div> <div>[yl]label</div> <div>[y2l]label</div> <div>[xu]nits</div> <div>[yu]nits</div> <div>[y2u]nits</div> <div>[t]itle</div> <div>[le]gend</div> <div>[all]</div> </div> <div> <div>hm</div> <div>hb</div> <div>cm</div> <div>cb</div> <div>tm</div> <div>tb</div> </div> <div> <div>8</div> <div>10</div> <div>12</div> <div>14</div> <div>18</div> <div>24</div> </div> <div>Colour</div> </div>	sets up fonts for the image:  fonts available: hm - helvetica medium cb - courier bold tm - times new roman medium etc...  font sizes in pt: 8, 10, 12 etc...for colours - see standard list below
Foreground	for	standard colour	for colours - see standard list below
Format style	fo	default automatic full	
Grid on	gr	fine normal bold heavy on or off	
Line thickness	li	on off fine normal bold heavy	Turn on plotting of curve lines Turn off plotting of curve lines (symbols drawn) set the line thickness to 1 pixel set the line thickness to 2 pixels set the line thickness to 4 pixels set the line thickness to 8 pixels
Model numbers on labels.	mn	auto on off	Adds a "model" prefix to the entity IS. If set to the "auto" only puts the model number on when there is more than 1 model in T/HIS
Model prefix format	prefix	id dir thf user	Set the model prefix to the model ID Set the model prefix to the job directory Set the model prefix to the base THF/model file Set the model prefix to the user defined one.
Reverse black white	re	on or off	
Size of plot	si	integer	xsize = value x 4, ysize = value x 3 (aspect fixed)

Solid x and y axis	so	on or off		
Symbols on	sy	on or off		
X grid controls	xau	-		
	xin	x grid increment		
	xoff	x grid offset		
Y grid controls	yau	-		
	yin	y grid increment		
	yoff	y grid offset		
Axis type, Linear/Logarithmic	xlin	-		Swap the x axis to a linear scale
	xlog	-		Swap the x axis to a logarithmic scale
	ylin	-		Swap the y axis to a linear scale
	ylog	-		Swap the y axis to a logarithmic scale
	y2lin	-		Swap the second y axis to a linear scale
	y2log	-		Swap the second y axis to a logarithmic scale
Axis display	axis	top	on or off	Turns ON/OFF display of graphs TOP axis
	axis	bottom	on or off	Turns ON/OFF display of graphs RIGHT axis

Plot setup description	plot setup word	following word(s)	notes
<b>Graph Title</b>	title	"title string"	Set the title for the graph.
	title_on	-	Turn on the display of the graph title
	title_off	-	Turn off the display of the graph title
<b>X Axis Properties</b>	x_lab	auto	Set the x axis label to be defined automatically
		manual	Set the x axis label to a user defined label
		"label string"	Set the user defined x axis label
		on	Turn on the display of the x axis label
		off	Turn off the display of the x axis label
	x_min	auto	Set the x axis minimum value to automatic
		numerical value	Set the x axis minimum value
	x_max	auto	Set the x axis minimum value to automatic
		numerical value	Set the x axis minimum value
	x_unit	auto	Set the x axis unit label to be defined automatically
		manual	Set the x axis unit label to a user defined label
		"unit string"	Set the user defined x axis label
		on	Turn on the display of the x axis unit label
		off	Turn off the display of the x axis unit label
<b>Y Axis Properties</b>	y_lab	auto	Set the y axis label to be defined automatically
		manual	Set the y axis label to a user defined label
		"label string"	Set the user defined y axis label
		on	Turn on the display of the y axis label
		off	Turn off the display of the y axis label
	y_min	auto	Set the y axis minimum value to automatic
		numerical value	Set the y axis minimum value
	y_max	auto	Set the y axis minimum value to automatic
		numerical value	Set the y axis minimum value
	y_unit	auto	Set the y axis unit label to be defined automatically
		manual	Set the y axis unit label to a user defined label
		"unit string"	Set the user defined y axis label
		on	Turn on the display of the y axis unit label
		off	Turn off the display of the y axis unit label

2nd Y Axis Properties	y2_lab	auto	Set the second y axis label to be defined automatically
		manual	Set the second y axis label to a user defined label
		"label string"	Set the user defined second y axis label
		on	Turn on the display of the second y axis label
		off	Turn off the display of the second y axis label
	y2_min	auto	Set the second y axis minimum value to automatic
		numerical value	Set the second y axis minimum value
	y2_max	auto	Set the second y axis minimum value to automatic
		numerical value	Set the second y axis minimum value
	y2_unit	auto	Set the second y axis unit label to be defined automatically
		manual	Set the second y axis unit label to a user defined label
		"unit string"	Set the user defined second y axis label
		on	Turn on the display of the second y axis unit label
		off	Turn off the display of the second y axis unit label

### 7.12.3.1 Deprecated Plot Setup Options

The following setup commands are still supported in Version 9.4 but they have been superseded by the new "properties" keyword (see section 7.10)

Plot setup description	plot setup word	following word(s)	notes
Set colour of min/max value	min_max	standard colour	for colours - see standard list below
Show max value	show	max	Turn on/off the highlight of the Maximum Value
Show max value	show	min	
Display X value at max	show	xmax	Display x value at Maximum
Display X value at min	show	xmin	Display x value at Maximum
Display Y value at max	show	ymax	Display y value at Maximum
Display Y value at min	show	ymin	Display y value at Maximum
Show 3ms Clip Widow	show	3ms	
Show HIC Widow	show	hic	

### 7.12.4 Curve Display

The list of curves displayed in each graph is controlled by the **display** keyword. The list of curves can contain a mixture of curve tags, curve numbers (prefixed with #) or curve groups. If curve tags are specified in the curve list then they can contain wildcards.

keyword	second word	notes
display	curve list	The curve list can contain a mixture of curve tags, curve numbers (prefixed with #) or curve groups. If curve tags are specified in the curve list then they can contain wildcards.

The following option can be appended to the **display** keyword after the curve list.

Additional format	format word	following word #1	following word #2	notes
Style application	sty	style name	-	Curves have styles applied in the order they were defined

In version 9.4 the the following additional options that can be appended to the **display** keyword after the curve list are still supported although there use is not recommended. Equivalent commands have been added to the [Plot Setup](#) commands along with a number of new options.

Additional format	format word	following word #1	following word #2	notes
Title	tit	title word #1	title word #2 etc	Takes following words as a title until another keyword is found
X axis options	xax	if numeric #1 - xaxis min	if numeric #2 - xaxis max	Takes following words as a label until another keyword is found
		otherwise xaxis label	otherwise xaxis label	

Y axis options	yax	if numeric #1 - yaxis min	if numeric #2 - yaxis max	Takes following words as a label until another keyword is found
		otherwise yaxis label	otherwise yaxis label	
2nd Y axis options	2ya	if numeric #1 - yaxis min	if numeric #2 - yaxis max	Takes following words as a label until another keyword is found
		otherwise yaxis label	otherwise yaxis label	

e.g. `display            curve_2`  
               `curve_1`

(display "curve\_1" and "curve\_2")

`display            &"Curve group 3"`  
               `curve_2`

`title SLED TEST \`  
`xax Time \`  
`yax Displacement`

(display "curve\_2" and all the curves in "Curve group 3". Set the plot title and x and y axis labels.)

## 7.12.5 Image Generation

Many different types of image format can be outputted from FAST-TCF.

In T/HIS 9.4 onwards the FAST-TCF image output options have been revised to allow multiple graphs and pages to be selected for output. The old pre 9.3 syntax (see Section 7.12.6) is still supported for existing scripts but users are strongly advised to move to the new command format where all options are prefixed with either the "display" or "image" keyword.

Description	keyword	following words
Image output	image	image options

The available image output options are

Option	keyword	format word	second word	third word	fourth word	notes
Bitmap (8 bit)	image	bit / bmp	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Bitmap (8 bit uncompressed)	image	bit_u / bmp_u	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Gif (8 bit)	image	gif	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Png (8 bit)	image	png	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Bitmap (24 bit)	image	bit24 / bmp24	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Pixel map (24 bit)	image	ppm / pix	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Jpeg (24 bit)	image	jpg / jpeg	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Png (24 bit)	image	png24	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
Postscript	image	ps	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'
PDF	image	pdf	filename	graph	all / active / 'n'	Generate an image containing all graphs / all active graphs / graph number 'n'
				page	all / current / 'n'	Generate an image for each page / the current page / page number 'n'

In addition to the image formats the following image output options can also be specified

Description	keyword	second word	notes
Image resolution	i_res	screen / 2x / 4x	Set the resolution to either the same as the screen or 2 or 4 times the screen resolution for image output
Postscript /PDF resolution	p_res	screen / 2x / 4x	Set the resolution to either the same as the screen or 2 or 4 times the screen resolution for Postscript and PDF output
Plot title	ti	title string	Specify the plot title (postscript / PDF output only)
Figure Number	fi	figure number	Specify the figure number (postscript / PDF output only)
Orientation	ori	land / port	Specify the paper orientation (postscript / PDF output only)

```
image bmp    output1.bmp graph all
```

(generate a bitmap called output1.bmp containing all the current graphs)

```
image jpeg   output2.jpg page 3
```

(generate a JPEG image called output2.jpg containing page 3)

```
image      2x
i_res
```

(set the resolution used for all following images to 2 x the screen resolution)

**image ti**      **Run number 2**

(set the plot title to "Run number 2" for any following postscript or PDF images)

**image ori**    **landscape**

(set the page layout to landscape for any following postscript or PDF images)

## 7.12.6 Pre 9.3 Image Output

The following pre T/HIS 9.3 image output commands are still supported but users are recommended to use the new format described above.

Curve styles that have been previously defined can be applied to the curves in the plot, and various other settings can be made with regards to the axes and titles.

Images that require a second yaxis need to determine which curves go on which axis. To do this, use a "##" in the curve listing to switch to the second axis. The options are described in the tables below.

Curve files can be included within the curves to plot. FAST-TCF detects a curve file to read in using the pattern string ".cur" at the end of the name. The curves are read in, styles are applied, and the image is plotted. The curves are then deleted.

The user can use wildcards ("\*") in the tag names to select multiple curves for plotting.

**bit d.bmp**      **#1 #3 CRV2 ## #2 #4 head\_accn**      **Title 2nd axis**  
**example**

(2 curves on 1st yaxis and 3 on 2nd yaxis)

(Title)

**bit h.bmp**      **#1 #3 CRV2 style ONE**      **xax 0 5E-3**      **Time**      **Title Head**

(curves)

(style to apply)

(xaxis min and max)

(XLabel)

(Title)

**bit l.bmp**      **#1 #100 reference.cur line.cur #1000**      **style**  
**reference**

(curves and curve files to plot)

(style to apply)

**bmp test.bmp accn\***

(all curves with tags beginning with "accn")

Description	keyword	second word	following words	following words
Bitmap	bit / bmp	filename	curve(s)	additional formatting
Bitmap (uncompressed)	bit_u / bmp_u	filename	curve(s)	additional formatting
Jpeg	jpg / jpeg	filename	curve(s)	additional formatting
Pixel map	ppm	filename	curve(s)	additional formatting
B & W postscript	post	filename	curve(s)	additional formatting
Colour postscript	cpost	filename	curve(s)	additional formatting

Additional format	format word	following word #1	following word #2	notes
Style application	sty	style name	-	Curves have styles applied in the order they were defined
Title	tit	title word #1	title word #2 etc	Takes following words as a title until another keyword is found
X axis options	xax	if numeric #1 - xaxis min otherwise xaxis label	if numeric #2 - xaxis max otherwise xaxis label	Takes following words as a label until another keyword is found
Y axis options	yax	if numeric #1 - yaxis min otherwise yaxis label	if numeric #2 - yaxis max otherwise yaxis label	Takes following words as a label until another keyword is found
2nd Y axis options	2ya	if numeric #1 - yaxis min otherwise yaxis label	if numeric #2 - yaxis max otherwise yaxis label	Takes following words as a label until another keyword is found

## 7.13 Outputting curve properties to text files, variables and REPORTER

These requests output a curve property (eg its maximum Y value) into a specified tabulation file, to a REPORTER variable in a text file, or into a variable within FAST-TCF.

Output type	keyword	2nd word	3rd word	4th word	extra words	Format (optional)	variable word	variable name	description words	Notes
Tabulation file	tab	filename	curve #	property to output	if values needed	format	varf	variable name	description	
Tabulation file append	taba	filename	curve #	property to output	if values needed	format	varf	variable name	description	
Tabulation file (csv)	tabc	filename	curve #	property to output	if values needed	format	varf	variable name	description	Each output is appended to the current line in the file.
Tabulation file (csv)	tabcr	filename	curve #	property to output	if values needed	format	varf	variable name	description	Each output is appended to the current line in the file, followed by a carriage return so that the next output starts a new line.
FAST-TCF variable	varf	variable name	curve #	property to output	if values needed	format	-	-	description	
REPORTER variable	var	variable name	curve #	property to output	if values needed	format	varf	variable name	description	
REPORTER variable append	vara	variable name	curve #	property to output	if values needed	format	varf	variable name	description	

### 7.13.1 Available Curve Properties

Various advanced requests can be performed (e.g. first non-zero Y, maximum in a window) and the table below describes them in more detail. Requests which require inputs (e.g. t1 and t2 of a window) take the default values in the table if the following words do not appear to be numbers, or if no following words exist.

Property to output	property word	value words	notes
Minimum x	minx	-	-
Maximum x	maxx	-	-
Minimum y	min	-	-
X at minimum y	xatmin	-	-
Y at minimum x	yatmin	-	-
Minimum y in window t1 t2	minw	t1 and t2	default t1=-1E19 and t2=+1E19
X at minimum y in window t1 t2	xminw	t1 and t2	default t1=-1E19 and t2=+1E19
Maximum y	max	-	-
X at maximum y	xatmax	-	-
Y at maximum x	yatmax	-	-
Maximum y in window t1 t2	maxw	t1 and t2	default t1=-1E19 and t2=+1E19
X at maximum y in window t1 t2	xmaxw	t1 and t2	default t1=-1E19 and t2=+1E19
Average in window t1 t2	ave	t1 and t2	default t1=-1E19 and t2=+1E19
Hic	hic	-	-
t1 of Hic window	hict1	-	-
t2 of Hic window	hict2	-	-
Hicd	hicd	-	-
t1 of Hicd window	hicdt1	-	-
t2 of Hicd window	hicdt2	-	-
3ms	3ms	-	-
t1 of 3ms window	3mst1	-	-
t2 of 3ms window	3mst2	-	-

Y at X	yatx	x value	default xvalue=-1E19
X when Y is passed after gate time	xygate	y value & gate time	default yvalue=-1E19, gate=+1E19
X at first non-zero Y	xnonz	-	nonzero = 1/1000000th of curve max
X at last non-zero Y	xfail	-	nonzero = 1/1000000th of curve max
Y value at last non-zero Y	yfail	-	nonzero = 1/1000000th of curve max
TTI	tti	-	-
Error Function - Max difference & time	max_err	-	-
Error Function - Difference as a %age of reference	pc_err	-	-
Error Function - Difference as a %age of peak reference	pc_max_err	-	-
Error Function - Average Difference	av_err	-	-
Error Function - Average Difference as a %age of peak reference	av_max_err	-	-
Error Function - Area weighted difference	area_err	-	-
Error Function - Max difference & time	err	-	-
Curve Correlation Function	correlate	-	Returns curve correlation value

## 7.13.2 Writing out curve properties to a text "tabulation" file

This is achieved using the "tabulation" command. This automatically overwrites any existing file in the output directory, but only on the **first occurrence** in the input script. If this is not desired then use the "taba" command which will append an existing file on the first tab call.

A 9.2 onwards option is the "**tabc**" command, which appends the data into csv format on the last line in the file. The first call to this command writes a **new line** to the file, and the subsequent calls append the end of this line. This enables the user to compare runs on a line by line basis in software such as Excel.

Some examples of writing out curve properties to a text file are below:

```
e.g.  tab output.txt #1          max          max y of curve #1
      (file output.txt)   (curve number)   (maximum Y)   (description)
      tab output.txt node_head_accn maxw          1.00E-03 30.00E-3
      (file output.txt)   (curve tag)       (max Y in window) (window t1) (window t2)
      taba output.txt node_head_accn min
      (append output.txt) (curve tag)       (minimum Y)
```

Properties for multiple curves can be output by specifying either multiple "tab" commands or by using a curve tag containing wildcards or a curve group.

```
e.g.  tab output.txt node_*          max          maximum y value
      (file output.txt)   (all curves with a tag starting with node_) (maximum Y)   (description)
      tab output.txt &group_1          max          maximum y value
      (file output.txt)   (all curves in group "group_1")   (maximum Y)   (description)
```

## 7.13.3 Writing out REPORTER variables

REPORTER can write curve properties to its reports, so FAST-TCF needs to output a text file that REPORTER can interrogate to find out the curve properties it needs. To tell FAST-TCF to output a REPORTER variable, the keyword "varr" is used (for backwards compatibility "var" is sufficient). Use "vara" to append to an existing file.

```
e.g.  varr head_hic          #1          hic          hic result for head node
      (REPORTER variable %head_hic%) (curve number 1) (output request) (description)
e.g.  vara max_y          #1          max          maximum y value
      (REPORTER variable %max_y%)   (curve number 1) (output request) (description)
```



## 7.13.4 Setting up new FAST-TCF variables to contain curve properties

If you wish to use a curve property as a new variable within FAST-TCF - there are two ways you can achieve this.

1. Use the keyword "varf". This should be used when the user doesn't also require the value to be outputted into a text file or a REPORTER file.
2. Within a "tab", "taba", "tabc" or "varr" line, use the word "varf" just **before** the description words. The variable name is defined as the word after "varf".

The variable value is equal to the return value of the request. The variable can then be used in any subsequent lines of FAST-TCF.

For instance, the simplest way to set the variable **MAX\_ACCN** to the max of curve #1 is:

```
varf MAX_ACCN #1 max
```

However, if the user wishes to combine writing a property to a text file and defining a variable in FAST-TCF, this syntax could be used:

```
tab output.txt #1 max varf MAX_ACCN
```

## 7.13.5 Format

From Version 9.3 onwards the format used to display the value can be controlled by adding an optional "format" keyword after the property to be output and any additional inputs that property requires. The format should be specified directly after the "format" keyword and should use standard "C" programming syntax to specify a floating point format using either f,e,E,g or G format specifiers.

```
e.g.  tab output.txt  head      max      max y of curve #1
      (file output.txt) (curve tag) (maximum Y) (description)

      tab output.txt  head      max      format %6.3f      max y of curve #1
      (file output.txt) (curve tag) (maximum Y) (format)      (description)

      tab output.txt  head      max      format %.3f      max y of curve #1
      (file output.txt) (curve tag) (maximum Y) (format)      (description)
```

Example formats

Number	Format	Output
12.3456	%5.2f	12.35
12.3456	%7.3e	1.2345e+01
12.3456	%7.3E	1.2345E+01
2345678.9	%.0f	2345678
2345678.9	%6.5g	2.3457e+06
2345678.9	%6.5G	2.3457E+06
-0.000013583	%4.3e	-1.358E-05

## 7.13.6 Description

From Version 9.3 onwards the description specified as part of the output for a curve property can contain the following keywords that will automatically be replaced with the corresponding curve property.

keyword	Curve Property
{tag}	FAST-TCF curve tag
{label}	Curve label
{id}	Entity ID that the curve was created from
{Model}	Model ID curve was created from

e.g. **tab output.txt head max Max accl of node {id}**  
(file output.txt) (curve tag) (maximum Y) (description)

**tab output.txt head max Model {model} max accl of node {id}**  
(file output.txt) (curve tag) (maximum Y) (description)

## 7.14 FAST-TCF CURVE OUTPUT

Curves can be written out to either T/HIS curve files or CSV files from within a FAST-TCF script by using either the "app", "cop", "csv" or "csv2" keyword.

Description	keyword	second word	third word		notes
Copy into file	cop	filename	curve list	-	will overwrite any previous file
Append into file	app	filename	curve list	-	will append any previous file
CSV file TYPE 1	csv	filename	curve list	-	will overwrite any previous csv file. CSV has the format X1,Y1,X2,Y2,X3,Y3
CSV file TYPE 2	csv2	filename	curve list	last word = auto	will overwrite any previous csv file. CSV has the format X1,Y1,Y2,Y3 x axis interval is taken from curve #1 if all curves are chosen
				2nd last word= x start time last word = x axis interval	will overwrite any previous csv file. CSV has the format X1,Y1,Y2,Y3 start time and interval are defined in the line

The curve list for all of these commands can contain either curve tags (with or without wildcards), curve numbers (prefixed with #), curve groups or '\*' to select all curves.

e.g. `copy output_file.cur curve_1 &"group 1"`

(Write "curve\_1" and all the curves in curve group "group 1" to a new file "output\_file.cur")

`append output_file.cur curve_1 &"group 1"`

(Append "curve\_1" and all the curves in curve group "group 1" to the file "output\_file.cur")

`csv output.csv curve_1* curve_2*`

(Write all curves with tags that start with "curve\_1" or "curve\_2" to a CSV called "output.csv")

**NOTE :** There is no limit to the number of curves that can be output to a file but there is a limit to the number of items that can be specified in the curve list (currently 100). If more than 100 curves are to be output to a file then a [curve group](#) containing all of the curves should be created and used within the curve list. Alternatively if the curves are being written to a T/HIS curve file then the first 100 curves can be output using the "cop" keyword and then additional curves can be appended to the file using the "app" keyword.

### 7.14.1 Specifying curves using Curve Numbers

When outputting curves to a file a range of explicit curve numbers can be specified using the syntax **#start:#end**. This option only applies to curve numbers because curve tag can be defined in an arbitrary order.

### 7.14.2 CSV files

If a CSV/CSV2 file is written out from within a FAST-TCF script then by default it will contain rows containing UNIT information for the curves if UNITS have been defined. This additional information can be turned off if it isn't required([see section 7.5.5](#))



# APPENDICES

[A LS-DYNA Data Components](#)

[B Format of a T/HIS Curve File](#)

[C Format of a T/HIS Bulk Data File](#)

[D Filtering](#)

[E Injury Criteria](#)

[F Curve Correlation](#)

[G The ERROR function](#)

[H The "preference" file](#)

[I Command line options and Windows file associations](#)

[J List of Typed Commands](#)

## APPENDIX A - LS-DYNA Data Components

### A.1 Model Data Components

The following global data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
DT	Time Step	yes		yes	yes
KE	Kinetic energy	yes		yes	yes
IE	Internal energy	yes		yes	yes
SWE	Stonewall energy			yes	yes
SPE	Spring and damper energy			yes	yes
HG	Hourglass energy	yes		yes	yes
SDE	System damping energy			yes	yes
JE	Joint internal energy			yes	yes
SIE	Sliding interface energy			yes	yes
EW	External work		yes	yes	yes
RBE	Rigid Body stopper energy			yes	
TE	Total energy	yes		yes	yes
TER	Total/initial energy ratio			yes	yes
VX	Average X velocity	yes		yes	yes
VY	Average Y velocity	yes		yes	yes
VZ	Average Z velocity	yes		yes	yes
TZC	Time per zone cycle			yes	yes
AM	Added mass			yes	yes
PM	%age Mass increase			yes	yes
EKE	Eroded Kinetic energy			yes	yes
EIE	Eroded Internal energy			yes	yes
EHG	Eroded Hourglass energy			yes	yes
ER	Energy Ratio w/o Eroded			yes	yes
DRCE	Current Distortional Kinetic Energy				yes
DRMX	Maximum Distortional Kinetic Energy				yes
DRCO	Convergence Factor				yes
DRKE	Total Kinetic Energy				yes

### A.2 Part Data Components

For Parts the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
KE	Kinetic energy	yes		yes	yes
IE	Internal energy	yes		yes	yes
HG	Hourglass energy	yes		yes	yes
TE	Total energy	yes		yes	yes

XM	X momentum			yes	yes
YM	Y momentum			yes	yes
ZM	Z momentum			yes	yes
VX	Average X velocity	yes		yes	yes
VY	Average Y velocity	yes		yes	yes
VZ	Average Z velocity	yes		yes	yes
MA	Mass	yes		yes	yes
EIE	Eroded Internal energy			yes	yes
ER	Energy Ratio w/o Eroded			yes	yes

## A.3 Part Group Data Components

For Part Groups the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
KE	Kinetic energy	yes		yes	yes
IE	Internal energy	yes		yes	yes
HG	Hourglass energy	yes		yes	yes
TE	Total energy	yes		yes	yes
MA	Mass	yes		yes	yes

## A.4 Nodal Data Components

For nodes the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
TE	Temperature	yes		yes	yes
DX	X Displacement	yes		yes	yes
DY	Y Displacement	yes		yes	yes
DZ	Z Displacement	yes		yes	yes
DM	Displacement Magnitude	yes		yes	yes
VX	X Velocity	yes		yes	yes
VY	Y Velocity	yes		yes	yes
VZ	Z Velocity	yes		yes	yes
VM	Velocity Magnitude	yes		yes	yes
AX	X Acceleration	yes		yes	yes
AY	Y Acceleration	yes		yes	yes
AZ	Z Acceleration	yes		yes	yes
AM	Acceleration Magnitude	yes		yes	yes
CX	X Co-ordinate			yes	yes
CY	Y Co-ordinate			yes	yes
CZ	Z Co-ordinate			yes	yes
RX	X Rotation			yes	yes
RY	Y Rotation			yes	yes
RZ	Z Rotation			yes	yes
RM	Rotation Magnitude			yes	yes

RVX	X Rotational Velocity			yes	yes
RVY	Y Rotational Velocity			yes	yes
RVZ	Z Rotational Velocity			yes	yes
RVM	Rotational Velocity Magnitude			yes	yes
RAX	X Rotational Acceleration			yes	yes
RAY	Y Rotational Acceleration			yes	yes
RAZ	Z Rotational Acceleration			yes	yes
RAM	Rotational Acceleration Magnitude			yes	yes
FLX	X Thermal Flux			yes	yes
FLY	Y Thermal Flux			yes	yes
FLZ	Z Thermal Flux			yes	yes
FLM	Thermal Flux Magnitude			yes	yes

## Frequency Domain Analysis

For a steady state dynamic analysis (SSD) the following nodal data components are available. For each data component both amplitude and phase angle data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
DX	X Displacement			yes	
DY	Y Displacement			yes	
DZ	Z Displacement			yes	
VX	X Velocity			yes	
VY	Y Velocity			yes	
VZ	Z Velocity			yes	
AX	X Acceleration			yes	
AY	Y Acceleration			yes	
AZ	Z Acceleration			yes	

For a random vibration analysis (PSD) the following nodal data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
DX	X Displacement	yes		yes	yes
DY	Y Displacement	yes		yes	yes
DZ	Z Displacement	yes		yes	yes
DM	Displacement Magnitude	yes		yes	yes
VX	X Velocity	yes		yes	yes
VY	Y Velocity	yes		yes	yes
VZ	Z Velocity	yes		yes	yes
VM	Velocity Magnitude	yes		yes	yes
AX	X Acceleration	yes		yes	yes
AY	Y Acceleration	yes		yes	yes
AZ	Z Acceleration	yes		yes	yes
AM	Acceleration Magnitude	yes		yes	yes

Only nodes that have been declared in "nodal time-history blocks" will be available for processing. To get a list of available node numbers in command line mode use the **M(enu)** command.



## Coordinate system of results

All nodal results are in the global cartesian coordinate system **except** at nodes which have been defined as accelerometers: these report accelerations in the local coordinate system of the accelerometer subject to any rotations its "parent" rigid body has undergone.

## A.5 Solid Data Components

For solids the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX	yes		yes	
SYX	Stress in YY	yes		yes	
SZZ	Stress in ZZ	yes		yes	
SXY	Stress in XY	yes		yes	
SYZ	Stress in YZ	yes		yes	
SZX	Stress in ZX	yes		yes	
SMX	Maximum Principal Stress	yes		yes	
SMN	Minimum Principal Stress	yes		yes	
SMS	Maximum Shear Stress	yes		yes	
SVM	Von Mises Stress	yes		yes	
SAV	Average Stress (Pressure)	yes		yes	
<b>Strain components</b>					
EFF	Effective Plastic Strain	yes		yes	
EXX	Strain in XX	yes		yes	
EYY	Strain in YY	yes		yes	
EZZ	Strain in ZZ	yes		yes	
EXY	Strain in XY	yes		yes	
EYZ	Strain in YZ	yes		yes	
EZX	Strain in ZX	yes		yes	
EMX	Maximum Principal Strain	yes		yes	
EMN	Minimum Principal Strain	yes		yes	
EMS	Maximum Shear Strain	yes		yes	
EVM	Von Mises Strain	yes		yes	
EAV	Average Strain	yes		yes	
<b>"Extra" components</b>					
SOEn	Extra Data Component	yes		yes	

## Frequency Domain Analysis

For a steady state dynamic analysis (SSD) the following nodal data components are available. For each data component both amplitude and phase angle data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX			yes	

SYX	Stress in YX			yes	
SZZ	Stress in ZZ			yes	
SXY	Stress in XY			yes	
SYZ	Stress in YZ			yes	
SZX	Stress in ZX			yes	

For a random vibration analysis (PSD) the following nodal data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX			yes	
SYX	Stress in YX			yes	
SZZ	Stress in ZZ			yes	
SXY	Stress in XY			yes	
SYZ	Stress in YZ			yes	
SZX	Stress in ZX			yes	

## Coordinate systems of results

The stress and strain tensors are reported in the global cartesian system unless the option to output results in the part coordinate system has been used. Writing the directional strain tensor is optional in LS-DYNA: it will only appear in the menu if it is present.

## "Extra" data components

The "extra" data components (**SOEn**) are also optional and only appear if present in the database. They are material dependent results, and are treated as scalar data of unknown type by T/HIS.

## A.6 Beam Data Components

For beams the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Basic data components</b>					
NX	Axial force	yes		yes	
NY	Shear force in Y	yes		yes	
NZ	Shear force in Z	yes		yes	
MY	Moment in Y	yes		yes	
MZ	Moment in Z	yes		yes	
MX	Torsional moment	yes		yes	
<b>"Plastic" data components</b>					
EAX	Axial strain	yes			
PE1	Plastic Bending energy : end 1	yes			
PE2	Plastic Bending energy : end 2	yes			
RY1	Y rotation : end 1	yes			
RY2	Y rotation : end 2	yes			
RZ1	Z rotation : end 1	yes			
RZ2	Z rotation : end 2	yes			

RX	Torsional rotation	yes			
MY1	Y Bending moment : end 1	yes			
MY2	Y Bending moment : end 2	yes			
MZ1	Z Bending moment : end 1	yes			
MZ2	Z Bending moment : end 2	yes			
ACE	Axial collapse energy	yes			
IE	Internal energy	yes			
<b>Integration Point Data</b>					
SXX	Axial stress	yes		yes	
SXY	XY Shear stress	yes		yes	
SZX	ZX Shear stress	yes		yes	
EFF	Effective plastic strain	yes			
EXX	Axial strain	yes		yes	
<b>Discrete Beams - Only available if DISBOUT ASCII file has bene written to LSDA (binout) file.</b>					
AXD	Axial displacement			yes	
YD	Displacement in Y			yes	
ZD	Displacement in Z			yes	
AXR	Axial rotation			yes	
YR	Rotation in Y			yes	
ZR	Rotation in Z			yes	

## Frequency Domain Analysis

For a steady state dynamic analysis (SSD) the following nodal data components are available. For each data component both amplitude and phase angle data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Basic data components</b>					
NX	Axial force			yes	
NY	Shear force in Y			yes	
NZ	Shear force in Z			yes	
MY	Moment in Y			yes	
MZ	Moment in Z			yes	
MX	Torsional moment			yes	

For a random vibration analysis (PSD) the following nodal data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Basic data components</b>					
NX	Axial force			yes	
NY	Shear force in Y			yes	
NZ	Shear force in Z			yes	
MY	Moment in Y			yes	
MZ	Moment in Z			yes	
MX	Torsional moment			yes	

## Additional Beam Results: written if requested from LS-DYNA

In addition to the basic data components additional beam results may be output to the **.THF** file for both Belytschko-Schwer and Hughes-Liu beam elements. As no indication of the element type is written to the **.THF** file it is impossible for T/HIS to work out whether a specific element is a Belytschko-Schwer or a Hughes-Liu beam. As the element type is unknown the user must know which element type a beam is in order to extract the correct results.

### Belytschko-Schwer Beams

If you have used Belytschko-Schwer beams with a resultant plastic material model the following "plastic" results will also be written out to **.THF** file: (Note that these data are written even if the **\*DATABASE EXTENT BINARY** card field **<beamip>** is not set - the presence of a resultant beam material triggers their output automatically. This is not the case for Hughes-Liu data components, for which output must be requested explicitly, see below.)

### Coordinate systems of results

Beam results are always output in the element local coordinate system. Only beams declared in "beam element time-history blocks" will be available.

### "Extra" data components

Where "extra" results are written, and T/HIS cannot resolve unambiguously whether they are Belytschko-Schwer plastic data, or Hughes-Liu stress/strain data, **it is your responsibility to interpret the results correctly.**

## A.7 Shell Data Components

For shells the following data components are available. These combine with directions for the data component, and in some cases a location through the shell thickness.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX	yes		yes	
SYX	Stress in YX	yes		yes	
SYY	Stress in YY	yes		yes	
SYZ	Stress in YZ	yes		yes	
SZZ	Stress in ZZ	yes		yes	
SZX	Stress in ZX	yes		yes	
SXY	Stress in XY	yes		yes	
SMX	Maximum Principal Stress	yes		yes	
SMN	Minimum Principal Stress	yes		yes	
SMS	Maximum Shear Stress	yes		yes	
SVM	Von Mises Stress	yes		yes	
SAV	Average Stress (Pressure)	yes		yes	
<b>Strain components</b>					
EFF	Effective Plastic Strain	yes		yes	
EEX	Strain in XX	yes		yes	
EYY	Strain in YY	yes		yes	
EZZ	Strain in ZZ	yes		yes	
EXY	Strain in XY	yes		yes	
EYZ	Strain in YZ	yes		yes	
EXZ	Strain in ZX	yes		yes	

EMX	Maximum Principal Strain	yes		yes	
EMN	Minimum Principal Strain	yes		yes	
EMS	Maximum Shear Strain	yes		yes	
EVM	Von Mises Strain	yes		yes	
EAV	Average Strain	yes		yes	
<b>Force / Moment components</b>					
MX	Moment in X	yes			
MY	Moment in Y	yes			
MXY	Moment in XY	yes			
QX	Shear force in X	yes			
QY	Shear force in Y	yes			
NX	Normal force in X	yes			
NY	Normal force in Y	yes			
NXY	Normal force in XY	yes			
<b>Miscellaneous components</b>					
T	Thickness	yes			
I	Internal energy density	yes			
<b>"Extra" components</b>					
An	Extra Data Component	yes		yes	

## Frequency Domain Analysis

For a steady state dynamic analysis (SSD) the following nodal data components are available. For each data component both amplitude and phase angle data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX			yes	
SYY	Stress in YY			yes	
SZZ	Stress in ZZ			yes	
SXY	Stress in XY			yes	
SYZ	Stress in YZ			yes	
SZX	Stress in ZX			yes	

For a random vibration analysis (PSD) the following nodal data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX			yes	
SYY	Stress in YY			yes	
SZZ	Stress in ZZ			yes	
SXY	Stress in XY			yes	
SYZ	Stress in YZ			yes	
SZX	Stress in ZX			yes	

### A.7.1 THF (d3thdt) File

Stress	Stress tensors are in the global cartesian system unless the option to use material axes has been invoked for orthotropic materials (CMPFLG on *DATABASE_EXTENT_BINARY). By default results are available at top and bottom integration points and mid-surface but values can be output for all through thickness integration points by using MAXINT on *DATABASE_EXTENT_BINARY
Strain	The Strain tensors output is optional. Values are in the global cartesian system unless the option to use material axes has been invoked for orthotropic materials (CMPFLG on *DATABASE_EXTENT_BINARY). Only values at the top and bottom integration points are output. T/HIS will average these values for the mid surface.
Forces & Moments	Force and moment resultants are <data> per unit width, and are written in the element local axis system. Refer to "Theory of Plates and Shells", Timoshenko, for a precise definition of these values.
Extra	The "Extra History" data components will only appear in the menu if they have been selected for output (NEIPS on *DATABASE_EXTENT_BINARY). These are output for the same surfaces / integration points as the stress tensor values.

## Through Thickness Integration Points (surfaces/layers)

NOTE The top and bottom "surfaces" are **not** the outer fibres if the default Gaussian integration rules are used, but : rather the outer and inner integration points. The relationship between integration point location and shell thickness depends on the number of integration points used.

The following diagram shows locations of integration points with respect to shell half-thickness ( $t/2$ ) assuming the default Gaussian integration rules have been used:

No of Points Distance of outer fibres from neutral axis as a proportion of  $t/2$

1	0.0 (membrane)
2	0.577 $t/2$
3	0.775
4	0.861 $t/2$
5	0.906

The "top" (or outer) point is on the positive local Z side of the element neutral axis. The output of shell data from LS-DYNA will fall into one of two categories, and the "surface" options available in T/HIS depend on this.

NOTE It is possible to use non-default integration schemes in LS-DYNA which may locate the integration points : at different places. This is an advanced topic: contact Oasys Ltd for advice.

### Default output case: 3 "surfaces"

In this case, regardless of how many integration points the shell elements may actually have through their thickness, LS-DYNA writes out:

Top surface :	Top integration point
Centre surface :	Computed neutral axis value
Bottom surface :	Bottom integration point

Note that the "centre" surface here is the neutral axis value. For membrane elements all three sets of values will be the same.

### Optional output case: user-defined number of integration points (maxint other than 0 or 3)

The number of through thickness integration points written to the THF file can be modified using the value of MAXINT on the \*DATABASE\_EXTENT\_BINARY card. If this parameter is changed then all thin and thick shell output written to the THF file will have MAXINT data "slots" for integration points in the file, regardless of how many integration points a given element may have through its thickness.

If MAXINT is not 3 then the order in which data is written to the THF file is controlled by the actual number of integration points of integration points in a shell's formulation. The following table illustrates output for the case of MAXINT not equal to 3

Data slot in file	Shell with 3 Integration points	Shell with 5 Integration points	Shell with any other number of integration points
#1	Middle	Middle	Bottom     Top
#2	Bottom	Bottom	
#3	Top	Bottom + 1	
#4	zero	Top - 1	
#5	zero	Top	
#6	zero	zero	

NOTE The THF file does NOT contain any information on the number of integration points each shell was defined with.

No explicit neutral axis value is calculated or output.

The outcome of writing more integration points than have been used in a shell formulation is undefined.

There is no guarantee that the "centre" surface in this context is the neutral axis value: this will depend upon the element integration scheme. In addition where the "centre" value has been averaged from a pair of points, when the number of layers is an even number, it will definitely not be the neutral axis value: consider plastic strain in a section in pure bending!

The ZTF file generated by PRIMER can help to resolve some of these problems.

## THF File + ZTF File

If a ZTF file had been generated using PRIMER then T/HIS can use additional information from the ZTF to correctly work out the number of integration points each shell element was defined with. If an attempt is made to output data for a surface that does not exist in the THF file then T/HIS will generate a warning message and a NULL curve will be generated.

In addition to working out the correct number of through thickness integration points for each element T/HIS can also use the information in the ZTF to identify models where MAXINT has been set to a -ve number in order to generate data for multiple in-plane integration points.

Effect of plotting "Top" surface on models with MAXINT = 6 and MAXINT = 9 with and without a ZTF file.

	MAXINT = 6, no ZTF file	MAXINT = 6, ZTF file present	MAXINT = 9, no ZTF file	MAXINT = 9, ZTF file present
Shell 1 has 4 integration points	Undefined (#int points < 6)	Correct (int point #4)	Undefined (#int points < 9)	Correct (int point #4)
Shell 2 has 6 integration points	Correct (int point #6)	Correct (int point #6)	Undefined (#int points < 9)	Correct (int point #6)
Shell 3 has 9 integration points	Incorrect (6th integration point)	Warning message as #int points < 6	Correct (int point #9)	Correct (int point #9)

## In-plane Integration Points

In some versions of LS-DYNA it is now possible to write out data for all 4 in-plane integration points for fully integrated shells by setting MAXINT on the \*DATABASE\_EXTENT\_BINARY card to a -ve number. For example specifying a value of -8 will generate data for 8 layers each with 4 in-plane integration points. If this option is used then all the elements will be written out using this option regardless of whether they are fully integrated or not.

As there is no information in the THF to indicate that data for 4 in-plane integration points has been written to the file then the file format will be exactly the same as for an analysis with a +ve value of MAXINT 4 times larger. For example MAXINT = -8 and MAXINT = 32 will both produce THF files with 32 integration points worth of data and there is no way for T/HIS to know which value of MAXINT was used to generate the data. The ZTF file generated by PRIMER can help to resolve this problem.

If multiple in-plane integration points are written to the THF file then they are written in the following order.

Layer 1 - in-plane int point #1  
 Layer 2 - in-plane int point #1  
 ....  
 Layer n - in-plane int point #1  
 Layer 1 - in-plane int point #2  
 Layer 2 - in-plane int point #2  
 ....  
 Layer n - in-plane int point #2  
 Layer 1 - in-plane int point #3  
 ....

**NOTE** If non fully integrated shells are included in the list of elements written to the THF file then in some versions of LS-DYNA the 2nd, 3rd and 4th in-plane values will all be zero. Care should therefore be taken if the 4 in-plane values are averaged.

In some versions of LS-DYNA the 1st in-plane integration point is correctly written out using the global axis system while the 2nd, 3rd and 4th in-plane values are written using the elements local coordinate system. Care should therefore be taken if the 4 in-plane values are averaged.

## A.7.2 LSDA (binout) File

Stress	By default stress tensors are in the local element coordinate system. Values are written out for all the through thickness and in-plane integration points.
Strain	The Strain tensors output is optional. By default the values are in the local element coordinate systems and only values at the top and bottom integration points are output. T/HIS will average these values for the mid surface.
Forces & Moments	These are not written to the LSDA file.
Extra	By default "Extra" data components are not written to the LSDA file.  Some recent versions of LS-DYNA can now write the "Extra" data components to the LSDA file if the parameters OPTION1, OPTION2, OPTION3 and OPTION4 are set on the *DATABASE_ELOUT card.

### Global v Local coordinate system results

The LSDA file can contain both ELOUT and ELOUTDET data components. By default T/HIS uses the data from ELOUTDET as ELOUT only contains a subset of the data in ELOUTDET.

In some versions of LS-DYNA it is possible to change the Shell and ThickShell data components written to the ELOUT so that they are defined using the Global coordinate system (see EOCS on \*CONTROL\_OUTPUT) instead of the default local element coordinate system. If this option is used then only the ELOUT file is modified, the ELOUTDET file is still written using the local element coordinate system.

If T/HIS detects that the LSDA file contains both ELOUT and ELOUTDET and that they are using different coordinate systems then T/HIS will display an additional option can be used to force T/HIS to use the ELOUT file data instead of the ELOUTDET data.

### Through Thickness Integration Points (surfaces/layers)

Unlike the THF file the LSDA file can contain different numbers of integration points for each element. This means that if "Top" surface is selected T/HIS can correctly identify which integration point it needs to read data from.

By default strain tensors are only written out for the top and bottom surfaces and T/HIS averages these for the mid surface values. In recent versions of LS-DYNA the parameter INTOUT on the \*DATABASE\_EXTENT\_BINARY card can change this so that strain tensor values are written out for all the through thickness integration points. T/HIS does not currently support these additional values.



## In-plane Integration Points

By default the LSDA file will contain data for all 4 in-plane integration points for any fully integrated shells. As with the THF file by default there is no information in the LSDA file to tell the difference between a shell with 32 through thickness integration points and a shell with 8 through thickness layers and 4 in-plane points per layer. If a ZTF file written by PRIMER is present then T/HIS can use the extra information on the ZTF to work out which elements have multiple in-plane points.

If the parameter INTOUT on the \*DATABASE\_EXTENT\_BINARY card is set then the format of the LSDA file is changed and the LSDA file then contains enough information for T/HIS to identify the shells with multiple in-plane integration points without the ZTF file.

In addition to changing the format of the LSDA file setting INTOUT on the \*DATABASE\_EXTENT\_BINARY card also outputs strain tensor values at each in-plane integration point as well as all the through thickness layers. T/HIS does not currently support strain values from multiple in-plane integration points.

## Extrapolated Stress / Strain Values

The parameter NODOUT on the \*DATABASE\_EXTENT\_BINARY card "gaa" can be used to generate stress and strain values that have been extrapolated to the nodal positions instead of values at the elements integration points. T/HIS does not currently support these extrapolated values.

## A.8 Thick Shell Data Components

For thick shells the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX	yes		yes	
SYX	Stress in YY	yes		yes	
SZZ	Stress in ZZ	yes		yes	
SXY	Stress in XY	yes		yes	
SYZ	Stress in YZ	yes		yes	
SZX	Stress in ZX	yes		yes	
SMX	Maximum Principal Stress	yes		yes	
SMN	Minimum Principal Stress	yes		yes	
SMS	Maximum Shear Stress	yes		yes	
SVM	Von Mises Stress	yes		yes	
SAV	Average Stress (Pressure)	yes		yes	
<b>Strain components</b>					
EFF	Effective Plastic Strain	yes		yes	
EXX	Strain in XX	yes		yes	
EYY	Strain in YY	yes		yes	
EZZ	Strain in ZZ	yes		yes	
EXY	Strain in XY	yes		yes	
EYZ	Strain in YZ	yes		yes	
EZX	Strain in ZX	yes		yes	
EMX	Maximum Principal Strain	yes		yes	
EMN	Minimum Principal Strain	yes		yes	

EMS	Maximum Shear Strain	yes		yes	
EVM	Von Mises Strain	yes		yes	
EAV	Average Strain	yes		yes	
<b>"Extra" components</b>					
An	Extra Data Component	yes		yes	

## Frequency Domain Analysis

For a steady state dynamic analysis (SSD) the following nodal data components are available. For each data component both amplitude and phase angle data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX			yes	
SYX	Stress in YX			yes	
SYY	Stress in YY			yes	
SYZ	Stress in YZ			yes	
SZZ	Stress in ZZ			yes	
SZX	Stress in ZX			yes	

For a random vibration analysis (PSD) the following nodal data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Stress components</b>					
SXX	Stress in XX			yes	
SYX	Stress in YX			yes	
SYY	Stress in YY			yes	
SYZ	Stress in YZ			yes	
SZZ	Stress in ZZ			yes	
SZX	Stress in ZX			yes	

### A.8.1 THF (d3thdt) File

Stress	Stress tensors are in the global cartesian system unless the option to use material axes has been invoked for orthotropic materials (CMPFLG on *DATABASE_EXTENT_BINARY). By default results are available at top and bottom integration points and mid-surface but values can be output for all through thickness integration points by using MAXINT on *DATABASE_EXTENT_BINARY
Strain	The Strain tensors output is optional. Values are in the global cartesian system unless the option to use material axes has been invoked for orthotropic materials (CMPFLG on *DATABASE_EXTENT_BINARY). Only values at the top and bottom integration points are output. T/HIS will average these values for the mid surface.
Extra	The "Extra History!" data components will only appear in the menu if they have been selected for output (NEIPS on *DATABASE_EXTENT_BINARY). These are output for the same surfaces / integration points as the stress tensor values.

## Through Thickness Integration Points (surfaces/layers)

NOTE The top and bottom "surfaces" are **not** the outer fibres if the default Gaussian integration rules are used, but rather the outer and inner integration points. The relationship between integration point location and shell thickness depends on the number of integration points used.

The following diagram shows locations of integration points with respect to shell half-thickness ( $t/2$ ) assuming the default Gaussian integration rules have been used:

No of Points Distance of outer fibres from neutral axis as a proportion of  $t/2$

1	0.0 (membrane)
2	0.577 $t/2$
3	0.775
4	0.861 $t/2$
5	0.906

The "top" (or outer) point is on the positive local Z side of the element neutral axis. The output of shell data from LS-DYNA will fall into one of two categories, and the "surface" options available in T/HIS depend on this.

**NOTE** It is possible to use non-default integration schemes in LS-DYNA which may locate the integration points at different places. This is an advanced topic: contact Oasys Ltd for advice.

### Default output case: 3 "surfaces"

In this case, regardless of how many integration points the shell elements may actually have through their thickness, LS-DYNA writes out:

Top surface :	Top integration point
Centre surface :	Computed neutral axis value
Bottom surface :	Bottom integration point

Note that the "centre" surface here is the neutral axis value. For membrane elements all three sets of values will be the same.

### Optional output case: user-defined number of integration points (maxint other than 0 or 3)

The number of through thickness integration points written to the THF file can be modified using the value of MAXINT on the \*DATABASE\_EXTENT\_BINARY card. If this parameter is changed then all thin and thick shell output written to the THF file will have MAXINT data "slots" for integration points in the file, regardless of how many integration points a given element may have through its thickness.

If MAXINT is not 3 then the order in which data is written to the THF file is controlled by the actual number of integration points of integration points in a shell's formulation. The following table illustrates output for the case of MAXINT not equal to 3

Data slot in file	Thick Shell with 3 Integration points	Thick Shell with any other number of integration points
#1	Middle	Bottom     Top
#2	Bottom	
#3	Top	
#4	zero	
#5	zero	
#6	zero	

**NOTE** The THF file does NOT contain any information on the number of integration points each shell was defined with.

No explicit neutral axis value is calculated or output.

The outcome of writing more integration points than have been used in a shell formulation is undefined.

There is no guarantee that the "centre" surface in this context is the neutral axis value: this will depend upon the element integration scheme. In addition where the "centre" value has been averaged from a pair of points, when the number of layers is an even number, it will definitely not be the neutral axis value: consider plastic strain in a section in pure bending!

The ZTF file generated by PRIMER can help to resolve some of these problems.

## THF File + ZTF File

If a ZTF file had been generated using PRIMER then T/HIS can use additional information from the ZTF to correctly work out the number of integration points each shell element was defined with. If an attempt is made to output data for a surface that does not exist in the THF file then T/HIS will generate a warning message and a NULL curve will be generated.

In addition to working out the correct number of through thickness integration points for each element T/HIS can also use the information in the ZTF to identify models where MAXINT has been set to a -ve number in order to generate data for multiple in-plane integration points.

Effect of plotting "Top" surface on models with MAXINT = 6 and MAXINT = 9 with and without a ZTF file.

	MAXINT = 6, no ZTF file	MAXINT = 6, ZTF file present	MAXINT = 9, no ZTF file	MAXINT = 9, ZTF file present
Thick Shell 1 has 4 integration points	Undefined (#int points < 6)	Correct (int point #4)	Undefined (#int points < 9)	Correct (int point #4)
Thick Shell 2 has 6 integration points	Correct (int point #6)	Correct (int point #6)	Undefined (#int points < 9)	Correct (int point #6)
Thick Shell 3 has 9 integration points	Incorrect (6th integration point)	Warning message as #int points < 6	Correct (int point #9)	Correct (int point #9)

## In-plane Integration Points

In some versions of LS-DYNA it is now possible to write out data for all 4 in-plane integration points for fully integrated shells by setting MAXINT on the \*DATABASE\_EXTENT\_BINARY card to a -ve number. For example specifying a value of -8 will generate data for 8 layers each with 4 in-plane integration points. If this option is used then all the elements will be written out using this option regardless of whether they are fully integrated or not.

As there is no information in the THF to indicate that data for 4 in-plane integration points has been written to the file then the file format will be exactly the same as for an analysis with a +ve value of MAXINT 4 times larger. For example MAXINT = -8 and MAXINT = 32 will both produce THF files with 32 integration points worth of data and there is no way for T/HIS to know which value of MAXINT was used to generate the data. The ZTF file generated by PRIMER can help to resolve this problem.

If multiple in-plane integration points are written to the THF file then they are written in the following order.

```

Layer 1 - in-plane int point #1
Layer 2 - in-plane int point #1
....
Layer n - in-plane int point #1
Layer 1 - in-plane int point #2
Layer 2 - in-plane int point #2
....
Layer n - in-plane int point #2
Layer 1 - in-plane int point #3
....

```

**NOTE** If non fully integrated shells are included in the list of elements written to the THF file then in some versions of LS-DYNA the 2nd, 3rd and 4th in-plane values will all be zero. Care should therefore be taken if the 4 in-plane values are averaged.

In some versions of LS-DYNA the 1st in-plane integration point is correctly written out using the global axis system while the 2nd, 3rd and 4th in-plane values are written using the elements local coordinate system. Care should therefore be taken if the 4 in-plane values are averaged.

## A.8.2 LSDA (binout) File

Stress	By default stress tensors are in the local element coordinate system. Values are written out for all the through thickness and in-plane integration points.
--------	---

Strain	The Strain tensors output is optional. By default values are in the local element coordinate systems and only values at the top and bottom integration points are output. T/HIS will average these values for the mid surface.
Extra	By default "Extra" data components are not written to the LSDA file.  Some recent versions of LS-DYNA can now write the "Extra" data components to the LSDA file if the parameters OPTION1, OPTION2, OPTION3 and OPTION4 are set on the *DATABASE_ELOUT card.

## Global v Local coordinate system results

The LSDA file can contain both ELOUT and ELOUTDET data components. By default T/HIS uses the data from ELOUTDET as ELOUT only contains a subset of the data in ELOUTDET.

In some versions of LS-DYNA it is possible to change the Shell and ThickShell data components written to the ELOUT so that they are defined using the Global coordinate system (see EOCS on \*CONTROL\_OUTPUT) instead of the default local element coordinate system. If this option is used then only the ELOUT file is modified, the ELOUTDET file is still written using the local element coordinate system.

If T/HIS detects that the LSDA file contains both ELOUT and ELOUTDET and that they are using different coordinate systems then T/HIS will display an additional option can be used to force T/HIS to use the ELOUT file data instead of the ELOUTDET data.

## Through Thickness Integration Points (surfaces/layers)

Unlike the THF file the LSDA file can contain different numbers of integration points for each element. This means that if "Top" surface is selected T/HIS can correctly identify which integration point it needs to read data from.

By default strain tensors are only written out for the top and bottom surfaces and T/HIS averages these for the mid surface values. In recent versions of LS-DYNA the parameter INTOUT on the \*DATABASE\_EXTENT\_BINARY card can change this so that strain tensor values are written out for all the through thickness integration points. T/HIS does not currently support these additional values.

## In-plane Integration Points

By default the LSDA file will contain data for all 4 in-plane integration points for any fully integrated shells. As with the THF file by default there is no information in the LSDA file to tell the difference between a shell with 32 through thickness integration points and a shell with 8 through thickness layers and 4 in-plane points per layer. If a ZTF file written by PRIMER is present then T/HIS can use the extra information on the ZTF to work out which elements have multiple in-plane points.

If the parameter INTOUT on the \*DATABASE\_EXTENT\_BINARY card is set then the format of the LSDA file is changed and the LSDA file then contains enough information for T/HIS to identify the shells with multiple in-plane integration points without the ZTF file.

In addition to changing the format of the LSDA file setting INTOUT on the \*DATABASE\_EXTENT\_BINARY card also outputs strain tensor values at each in-plane integration point as well as all the through thickness layers. T/HIS does not currently support strain values from multiple in-plane integration points.

## Extrapolated Stress / Strain Values

The parameter NODOUT on the \*DATABASE\_EXTENT\_BINARY card can be used to generate stress and strain values that have been extrapolated to the nodal positions instead of values at the elements integration points. T/HIS does not currently support these extrapolated values.

## A.9 Rigid Wall Data Components

For rigid walls the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FN	Normal force		yes	yes	yes
FX	Global X force			yes	yes
FY	Global Y force			yes	yes
FZ	Global Z force			yes	yes

## A.10 Discrete Element (Spring/Damper) Data Components

For springs and dampers the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FT	Force		yes	yes	yes
ET	Elongation		yes	yes	yes
FE	Force versus Elongation		yes		
EN	Energy		yes		
MT	Moment		yes	yes	yes
RT	Rotation		yes	yes	yes
MR	Moment versus Rotation		yes		
FX	Global X force			yes	yes
FY	Global Y force			yes	yes
FZ	Global Z force			yes	yes
MX	Moment in X			yes	yes
MY	Moment in Y			yes	yes
MZ	Moment in Z			yes	yes

## A.11 Seat Belt Data Components

For seat belts the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FT	Force		yes	yes	yes
ST	Strain		yes		
FS	Force versus Strain		yes		
CL	Current Length			yes	yes

## A.12 Retractor Data Components

For retractors the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FT	Force		yes	yes	yes
PT	Pullout		yes	yes	yes
FP	Force versus Pullout		yes		

## A.13 Slipping Data Components

For slippings the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
PT	Pull through		yes	yes	yes

## A.14 Contact Data Components

For contacts the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FX	Master X force		yes	yes	yes
FY	Master Y force		yes	yes	yes
FZ	Master Z force		yes	yes	yes
FM	Master Force Magnitude		yes	yes	yes
FXS	Slave X force		yes	yes	yes
FYS	Slave Y force		yes	yes	yes
FZS	Slave Z force		yes	yes	yes
FMS	Slave Force Magnitude		yes	yes	yes
TEN	Total energy (Slave + Master)		yes	yes	yes
SEN	Slave side energy		yes	yes	yes
MEN	Master side energy		yes	yes	yes
FRI	Frictional energy		yes	yes	yes

## A.15 Nodal Reaction Force Data Components

For nodal reactions the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FX	X Force		yes	yes	yes
FY	Y Force		yes	yes	yes
FZ	Z Force		yes	yes	yes
FM	Force Magnitude		yes	yes	yes
EN	Energy			yes	yes
LFX	Local X force			yes	yes
LFY	Local Y force			yes	yes
LFZ	Local Z force			yes	yes

## A.16 Airbag Data Components

For airbags the following data components are available. Versions of LS-DYNA 971 can also generate PART based data for AIRBAGS that use the PARTICLE airbag methods.

If \*DATABASE\_CPM\_SENSOR has been used to define sensors then the output for the sensors will also be available under the AIRBAG data components.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Airbag components</b>					
PR	Pressure		yes	yes	yes
VO	Volume		yes	yes	yes
IE	Internal energy		yes	yes	yes

IN	Mass flow rate in		yes	yes	yes
OU	Mass flow rate out		yes	yes	yes
TM	Total mass		yes	yes	yes
DE	Density			yes	yes
SA	Surface area			yes	yes
TE	Gas temperature			yes	yes
RF	Reaction force			yes	yes
<b>Part components</b>					
PR	Pressure			yes	
MAF	Flow rate through fabric			yes	
MAV	Flow rate through vent			yes	
TA	Total area			yes	
UN	Unblocked area			yes	
TE	Gas temperature			yes	
<b>Airbag Chamber components</b>					
PR	Pressure			yes	
VO	Volume			yes	
IE	Internal energy			yes	
IN	Mass flow rate in			yes	
OU	Mass flow rate out			yes	
TM	Total mass			yes	
DE	Density			yes	
SA	Surface area			yes	
TE	Gas temperature			yes	
RF	Reaction force			yes	

**CPM Sensor Components (\*DATABASE\_CPM\_SENSOR)**

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
X	X Co-ordinate of Sensor			yes	yes
Y	Y Co-ordinate of Sensor			yes	yes
Z	Z Co-ordinate of Sensor			yes	yes
VX	X Velocity			yes	yes
VY	Y Velocity			yes	yes
VZ	Z Velocity			yes	yes
VM	Velocity Magnitude			yes	yes
PR	Pressure			yes	yes
DE	Density			yes	yes
TE	Temperature			yes	yes

## A.17 Joint Data Components

For joints the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
<b>Basic Joints</b>					



FX	Global X force			yes	yes
FY	Global Y force			yes	yes
FZ	Global Z force			yes	yes
FM	Force Magnitude			yes	yes
MX	Moment in X			yes	yes
MY	Moment in Y			yes	yes
MZ	Moment in Z			yes	yes
MM	Moment Magnitude			yes	yes
<b>General Stiffness Joints</b>					
PHA	Phi angle			yes	yes
DPH	d(Phi)/dt			yes	yes
PHS	Phi stiffness moment			yes	yes
PHD	Phi damping moment			yes	yes
PHT	Phi total moment			yes	yes
THA	Theta angle			yes	yes
DTH	d(Theta)/dt			yes	yes
THS	Theta stiffness moment			yes	yes
THD	Theta damping moment			yes	yes
THT	Theta total moment			yes	yes
PSA	Psi angle			yes	yes
DPS	d(Psi)/dt			yes	yes
PSS	Psi stiffness moment			yes	yes
PSD	Psi damping moment			yes	yes
PST	Psi total moment			yes	yes
EN	Total joint energy			yes	yes
<b>Flexion Torsion Joints</b>					
AA	Alpha angle			yes	yes
DA	d(Alpha)/dt			yes	yes
ALS	Alpha stiffness moment			yes	yes
ALD	Alpha damping moment			yes	yes
ALT	Alpha total moment			yes	yes
BA	Beta angle			yes	yes
DB	d(Beta)/dt			yes	yes
BES	Beta stiffness moment			yes	yes
BED	Beta damping moment			yes	yes
BET	Beta total moment			yes	yes
GA	Gamma angle			yes	yes
DG	d(Gamma)/dt			yes	yes
GSF	Gamma scale factor			yes	yes
EN	Total joint energy			yes	yes
<b>Translational Joints</b>					
XD	X displacement			yes	yes
DXD	d(X)/dt			yes	yes
YD	Y displacement			yes	yes

DYD	$d(Y)/dt$			yes	yes
ZD	Z displacement			yes	yes
DZD	$d(Z)/dt$			yes	yes
XSF	X stiffness			yes	yes
XDF	X damping			yes	yes
XTF	X total			yes	yes
YSF	Y stiffness			yes	yes
YDF	Y damping			yes	yes
YTF	Y total			yes	yes
ZSF	Z stiffness			yes	yes
ZDF	Z damping			yes	yes
ZTF	Z total			yes	yes
EN	Total joint energy			yes	yes

## A.18 Cross Section Data Components

For cross sections the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FX	X Force			yes	yes
FY	Y Force			yes	yes
FZ	Z Force			yes	yes
RM	Force Magnitude			yes	yes
MX	Moment in X			yes	yes
MY	Moment in Y			yes	yes
MZ	Moment in Z			yes	yes
MM	Moment Magnitude			yes	yes
CX	X centroid coordinate			yes	yes
CY	Y centroid coordinate			yes	yes
CZ	Z centroid coordinate			yes	yes
AR	Area of Cross Section			yes	yes

## A.19 Subsystem Data Components

For subsystems the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
KE	Kinetic energy			yes	yes
IE	Internal energy			yes	yes
HG	Hourglass energy			yes	yes
KR	Kinetic energy ratio			yes	yes
IM	Internal energy ratio			yes	yes
XM	X momentum			yes	yes
YM	Y momentum			yes	yes
ZM	Z momentum			yes	yes

## A.20 Geometric Contact Data Components

For geometric contact entities the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FX	X Force			yes	yes
FY	Y Force			yes	yes
FZ	Z Force			yes	yes
RM	Force Magnitude			yes	yes
MX	Moment in X			yes	yes
MY	Moment in Y			yes	yes
MZ	Moment in Z			yes	yes
MM	Moment Magnitude			yes	yes

## A.21 Nodal Rigid Body Data Components

For nodal rigid bodies the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
DX	X Displacement			yes	yes
DY	Y Displacement			yes	yes
DZ	Z Displacement			yes	yes
DM	Displacement Magnitude			yes	yes
VX	X Velocity			yes	yes
VY	Y Velocity			yes	yes
VZ	Z Velocity			yes	yes
VM	Velocity Magnitude			yes	yes
AX	X Acceleration			yes	yes
AY	Y Acceleration			yes	yes
AZ	Z Acceleration			yes	yes
AM	Acceleration Magnitude			yes	yes
CX	X Co-ordinate			yes	yes
CY	Y Co-ordinate			yes	yes
CZ	Z Co-ordinate			yes	yes
RX	X Rotation			yes	yes
RY	Y Rotation			yes	yes
RZ	Z Rotation			yes	yes
RM	Rotation Magnitude			yes	yes
RVX	X Rotational Velocity			yes	yes
RVY	Y Rotational Velocity			yes	yes
RVZ	Z Rotational Velocity			yes	yes
RVM	Rotational Velocity Magnitude			yes	yes
RAX	X Rotational Acceleration			yes	yes
RAY	Y Rotational Acceleration			yes	yes
RAZ	Z Rotational Acceleration			yes	yes
RAM	Rotational Acceleration Magnitude			yes	yes
D11	Direction Cosine 11			yes	
D12	Direction Cosine 12			yes	
D13	Direction Cosine 13			yes	
D21	Direction Cosine 21			yes	
D22	Direction Cosine 22			yes	
D23	Direction Cosine 23			yes	
D31	Direction Cosine 31			yes	
D32	Direction Cosine 32			yes	
D33	Direction Cosine 33			yes	
LDX	Local X Displacement			yes	yes
LDY	Local Y Displacement			yes	yes
LDZ	Local Z Displacement			yes	yes

LVX	Local X Velocity			yes	yes
LVY	Local Y Velocity			yes	yes
LVZ	Local Z Velocity			yes	yes
LAX	Local X Acceleration			yes	yes
LAY	Local Y Acceleration			yes	yes
LAZ	Local Z Acceleration			yes	yes
LRX	Local X Rotation			yes	yes
LRY	Local Y Rotation			yes	yes
LRZ	Local Z Rotation			yes	yes
LRVX	Local X Rotational Velocity			yes	yes
LRVY	Local Y Rotational Velocity			yes	yes
LRVZ	Local Z Rotational Velocity			yes	yes
LRAX	Local X Rotational Acceleration			yes	yes
LRAY	Local Y Rotational Acceleration			yes	yes
LRAZ	Local Z Rotational Acceleration			yes	yes

## A.22 Spotweld Data Components

For spotwelds the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
AX	Axial force			yes	yes
SH	Shear force			yes	yes
LE	Length			yes	yes
FT	Failure Time			yes	yes
FA	Failure			yes	yes
MM	Moment Magnitude			yes	yes
TO	Torsion			yes	yes

The following additional data components are also available for Solid Spotwelds and Spotweld Assemblies if the DCFAIL file is written.

FF	DC Failure Function			yes	yes
NF	Normal Failure Term			yes	yes
SF	Shear Failure Term			yes	yes
BF	Bending Failure Term			yes	yes
AR	Spotweld Area			yes	yes

The DCFAIL file contains additional data for spotweld solids and clusters models using the \_DAIMLERCHRYSLER version of \*MAT\_SPOTWELD (this version of the material does not support beam elements). The file contains additional failure data showing how close to failure the spotweld is in tension, shear, bending and torsion, in addition it contains another copy the normal spotweld forces written to the SWFORC file.

The new data components appear under the SOLID and ASSEMBLY sub types within the SPOTWELD menu. If the SWFORC file is also present then the normal forces and read from the SWFORC file, if the SWFORC file doesn't exist but the DCFAIL file does then the data components (Normal, shear forces etc) that are mirrored in the DCFAIL file are read from there instead.

As the DCFAIL file only contains the ID's and not the types or each connection then it is not possible to tell from the DCFAIL file alone which items are solids and which ones are spotweld clusters. If the SWFORC file is present then T/HIS used the information in this file to match up the ID's and work out the type of each item in the DCFAIL file. If the SWFORC file isn't present then it attempts to use the data in the ZTF file to work out the types.

## A.23 SPC Data Components

For SPC's the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FX	X Force			yes	yes
FY	Y Force			yes	yes
FZ	Z Force			yes	yes
FM	Force Magnitude			yes	yes
MX	Moment in X			yes	yes
MY	Moment in Y			yes	yes
MZ	Moment in Z			yes	yes
MM	Moment Magnitude			yes	yes

## A.24 Boundary Condition Data Components

For SPC's the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
For Pressure and Force Boundary conditions the following components are available.					
FX	Applied X Force			yes	yes
FY	Applied Y Force			yes	yes
FZ	Applied Z Force			yes	yes
FR	Applied Resultant force			yes	yes
EN	Energy from applied force			yes	yes
For Nodal Velocity Boundary conditions the following components are available.					
FX	Boundary condition motion X Force			yes	yes
FY	Boundary condition motion Y Force			yes	yes
FZ	Boundary condition motion Z Force			yes	yes
FR	Resultant Boundary condition motion force			yes	yes
EN	Energy from Boundary condition motion			yes	yes
For Rigid Body Velocity Boundary conditions the following components are available.					
FX	Boundary condition motion X Force			yes	yes
FY	Boundary condition motion Y Force			yes	yes
FZ	Boundary condition motion Z Force			yes	yes
FR	Resultant Boundary condition motion force			yes	yes
EN	Energy from Boundary condition motion			yes	yes
MX	Boundary condition motion X Moment			yes	yes
MY	Boundary condition motion Y Moment			yes	yes
MZ	Boundary condition motion Z Moment			yes	yes
MM	Boundary condition moment Magnitude			yes	yes

## A.25 FSI Data Components

For Fluid structural interactions the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
PR	Pressure				yes
FX	X Force				yes
FY	Y Force				yes
FZ	Z Force				yes
FM	Force Magnitude				yes
PL	Porous Leakage				yes
MF	Mass Flux				yes
LFX	Leakage X Force				yes
LFY	Leakage X Force				yes
LFZ	Leakage X Force				yes
LFM	Leakage Force Magnitude				yes
TE	Part Temperature				yes
X	X Co-ordinate of Sensor				yes
Y	Y Co-ordinate of Sensor				yes
Z	Z Co-ordinate of Sensor				yes
PR	Pressure				yes
SO	Cpld Solid ID				yes
TE	Temperature at Sensor				yes

## A.26 SPH Data Components

For SPH's the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
DE	Density			yes	yes
EXX	Strain in XX			yes	yes
EYY	Strain in YY			yes	yes
EZZ	Strain in ZZ			yes	yes
EXY	Strain in XY			yes	yes
EYZ	Strain in YZ			yes	yes
EZX	Strain in ZX			yes	yes
EFS	Effective Stress			yes	yes
SXX	Stress in XX			yes	yes
SYY	Stress in YY			yes	yes
SZZ	Stress in ZZ			yes	yes
SXY	Stress in XY			yes	yes
SYZ	Stress in YZ			yes	yes
SZX	Stress in ZX			yes	yes
SM	Smoothing Length			yes	yes

## A.27 TRACER Data Components

For TRACERs the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
CX	X Co-ordinate			yes	yes
CY	Y Co-ordinate			yes	yes
CZ	Z Co-ordinate			yes	yes
CV	Current vector			yes	yes
VX	X Velocity			yes	yes
VY	Y Velocity			yes	yes
VZ	Z Velocity			yes	yes
VM	Velocity Magnitude			yes	yes
SXX	Stress in XX			yes	yes
SYX	Stress in YY			yes	yes
SZZ	Stress in ZZ			yes	yes
SXY	Stress in XY			yes	yes
SYZ	Stress in YZ			yes	yes
SZX	Stress in ZX			yes	yes
EFP	Effective Plastic Strain			yes	yes
DE	Density			yes	yes
RV	Relative volume			yes	yes
AC	Active			yes	yes

## A.28 PULLEY Data Components

For PULLEYs the following data components are available.

	Component	THF (d3thdt)	XTF (xtfile)	LSDA (binout)	ASCII
FT	Force			yes	yes
SL	Slip			yes	yes
SR	Slip Rate			yes	yes
AN	Warp Angle			yes	yes



## APPENDIX B - T/HIS CURVE FILE FORMAT

A curve file is a file of x, y values which can be read into T/HIS for plotting. It can be written by T/HIS or by another program, or created using a text editor.

The format is as flexible as possible to allow many types of data to be handled.

```

Line 1      : Title
Line 2      : X axis label
Line 3      : Y axis label
Line 4      : Curve label
Line 5      : X, Y point 1
Line 6      : X, Y point 2
:           :
Line n+4    : X, Y point n

```

The X and Y values can be in any format as long as the two values are separated by either a space or comma. Up to 500000 points can be input.

Several curves can be put in one file sequentially, separated by the word CONTINUE. The title and three label lines must be present for each curve.

A comment line may be included anywhere in the file by starting the line with a '\$'.

Comment lines above the curve's title can contain styles and curve tags associated with the corresponding curve.

### B.1 Curve STYLE Information

From version 9.1 onwards T/HIS will recognise a line starting **\$ STYLE** as a style request for the following curve and the curve will be displayed with the corresponding style

A **\$ STYLE** line will take the format

**\$ STYLE : LINE STYLE, LINE COLOUR, LINE WIDTH, LINE SYMBOLS, SYMBOL FREQUENCY**

The following **\$ STYLE** options are available:

Style options	Available styles	Default
<b>LINE STYLE</b>	solid dash none	solid
<b>LINE COLOUR</b>	white red green blue cyan magenta yellow orange turquoise indigo lime	dependent on curve#
<b>LINE WIDTH</b>	fine normal bold heavy	normal

<b>LINE SYMBOLS</b>	triangle square diamond hourglass cross circle start dot null	dependent on curve#
<b>SYMBOL FREQUENCY</b>	frequency number (integer)	

## B.2 Curve TAGs

T/HIS will recognise a line starting with **\$ TAG** as a tag for the following curve and the tag can be used in T/HIS to reference the corresponding curve

a **\$ TAG** line will take the format

**\$ TAG : tag name**

## B.3 Curve UNITS

From version 9.4 onwards a T/HIS curve file can also contain information on the Unit system and the X and Y axis units.

A unit system is defined by a line starting with **\$ UNIT SYSTEM** and will take the format

**\$ UNIT SYSTEM : system name**

The following unit systems names can be specified by using either the full name or just "Un."

U1: m,kg,s (SI)  
U2: mm,T,s  
U3: mm,kg,ms  
U4: mm,gm,ms  
U5: ft,slug,s  
U6: m,T,s

The X and Y axis units are defined by a line starting with either **\$ X AXIS UNIT** or **\$ Y AXIS UNIT** and take one of the 2 following formats

**\$ X AXIS UNIT : unit name**  
**\$ X AXIS UNIT : mass,length,time,angle,temperature**

For the 1st format the following predefined unit names are available.

<b>Time</b>	<b>Rotation</b>	<b>Momentum</b>	<b>Energy Den</b>
<b>Energy</b>	<b>Rot Vel</b>	<b>Density</b>	<b>Mass Flow</b>
<b>Work</b>	<b>Rot Accel</b>	<b>Stress</b>	<b>Frequency</b>
<b>Temperature</b>	<b>Length</b>	<b>Strain</b>	<b>Power</b>
<b>Displacement</b>	<b>Area</b>	<b>Force</b>	<b>Thermal Flux</b>
<b>Velocity</b>	<b>Volume</b>	<b>Moment</b>	<b>Force width</b>
<b>Accel</b>	<b>Mass</b>	<b>Pressure</b>	<b>Moment width</b>

If the axis units are NOT one of these predefined units then the second input format can be used to define the unit in terms of it's basic properties. The values for **mass,length,time,angle** and **temperature** should be the powers that are used to describe the unit in terms of it's fundamental dimensions.

Some examples of common units defined using this method are shown below.

<b>Unit</b>	<b>Mass</b>	<b>Length</b>	<b>Time</b>	<b>Angle</b>	<b>Temperature</b>
Time	0.0	0.0	1.0	0.0	0.0
Displacement	0.0	1.0	0.0	0.0	0.0
Velocity	0.0	1.0	-1.0	0.0	0.0
Acceleration	0.0	1.0	-2.0	0.0	0.0
Stress	-1.0	1.0	-2.0	0.0	0.0

## B.4 Example

The following example shows a curve file containing 2 curves.

The first curve will be plotted with a bold, solid, green line with triangular symbols every other data point. The curve contains 5 data points and is given a reference tag CURVE\_1

The second curve will be plotted with a dashed, white, normal line. No symbols will be displayed. The curve contains 2 data points and has no reference tag.

\$		Comment line
\$ STYLE : solid,green,bold,triangle,2		Style line
\$ TAG : CURVE_1		Tag line
\$		Comment line
CURVE FILE EXAMPLE		;Title
Time		;X axis label
Displacement		;Y axis label
Curve number 1		;Curve label
0	2.0	;1st data pair
1.0	4E-3	
4.0,	4.7	
5 4		
10.0	8.9	
CONTINUE		;End of 1st curve
\$		Comment line
\$		Comment line
\$ STYLE : dash,white,,,		Style line
CURVE FILE EXAMPLE		;Title
Time		
Displacement		
Curve number 2		
0.0	7E2	
2.0	8.7E-9	
Notes:		

The abscissa (x axis) values are assumed to be in the correct order.

The free format allowed for the data points

The style line must contain 5 comma separated words in the order LINE STYLE, LINE COLOUR, LINE WIDTH, LINE SYMBOLS, SYMBOL FREQUENCY to be successfully understood by T/HIS

If any words are unspecified in the style line, as in curve 2, T/HIS will take the default option



## APPENDIX C - T/HIS BULK DATA FILE FORMAT

Format of a T/HIS Bulk Data File.

A bulk data file contains a number of curves that share the same X values.

The format of the file is as follows:

```
Line 1      :   Title
Line 2      :   Number of curves (maximum 12)
Line 3      :   Format, see Note 1 below
Line 4      :   Multipliers on values, see Note 2 below
Line 5      :   Axis labels, see Note 3 below
Line 6      :   Line labels, see Note 4 below
Line 7      :   X, Y1, Y2, Y3 ..... point 1
Line 8      :   X, Y1, Y2, Y3 ..... point 2
Line n+6    :   X, Y1, Y2, Y3 ..... point n
```

Up to 500000 points can be read in for each curve.

Note 1 The format for the point data must be given as a standard Fortran format statement, for example (F10.3, 4F10.2). The external brackets around the format must be included. If the data can be read in as a free format then type **FREE** or leave this line blank. Note however, free data is read in more slowly than formatted data.

Note 2 The multipliers are the amount by which the values read in are to be multiplied. For example you may wish to correct from ms to s or units of **G** (gravity) to mm/s<sup>2</sup>. On this line give the multipliers in the order X-value, Y1-value, Y2-value, etc. Separate each multiplier by a space or comma. A zero value is assumed to be 1. If all curves are to be read in as defined leave this line blank.

Note 3 The axis labels are character strings, separated by commas given in the following order.  
X-axis label, Y1-axis label, Y2-axis label, etc.

Note 4 The line labels are character strings separated by commas given in the following order.  
Line label 1, Line label 2, Line label 3, etc.

A comment line may be included any where in the file by starting the line with a \$.

The following shows a bulk data file with three curves and seven points on each curve.

```
$ Comment line
Title of the curves
3
FREE
$ A multiplier of 10 on X values and 5 on Y2 values
10,,5,
x-axis,y1-axis,y2-axis,y3-axis
curve 1,curve 2,curve 3
$ Now for the data
0.0 0.0 1.0 2.0
1.0 1.0 3.0 4.0
2.0 2.0 4.0 5.0
2.4 4.4 5.5 7.4
3.3 7.8 5.8 9.2
4.4 10.0 12.0 13.0
```



## APPENDIX D - FILTERING

This Appendix describes the filtering options within T/HIS.

Curves can be filtered to remove high frequency noise. The technique is typically applied to acceleration and force traces. Options available include standard filters (Channel Frequency Classes 60, 180, 600 and 1000 as per British Standard BS AU 228: Part 1: 1989, and the USA's National Highway Traffic Safety Administration (NHTSA) FIR filter). The standard filters (except the FIR filter) are all special cases of the Butterworth filter.

### D.1 Curve Regulation

All filtering options require the curves to have a constant time increment between points. This will generally be the case if the curves are LS-DYNA time history results. If not, the REGULARISE option will convert the curve to constant time increment.

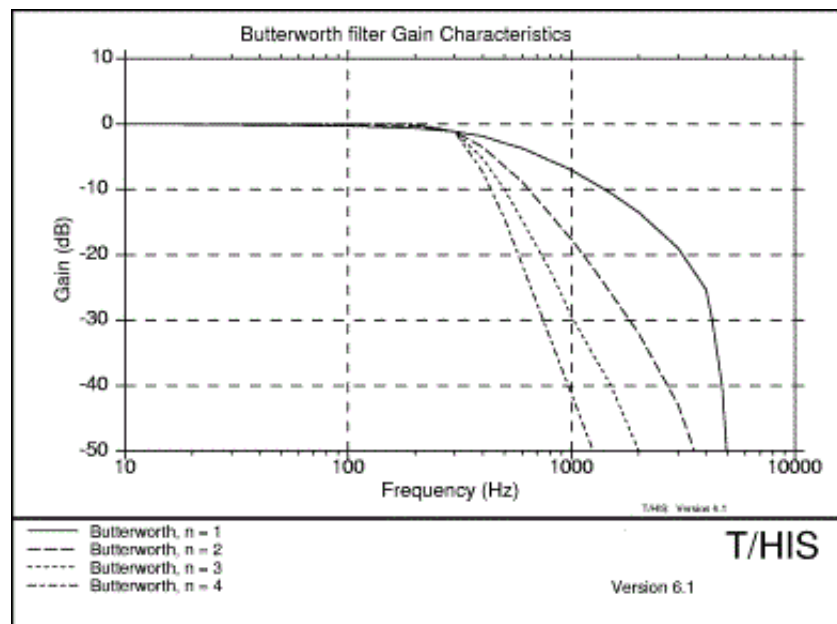
Typically the time increment should be at least 10 times the cut-off frequency; 10kHz (a 0.0001 second interval time base) is a good choice for automotive crash applications.

### D.2 Use of the Butterworth Filter Option

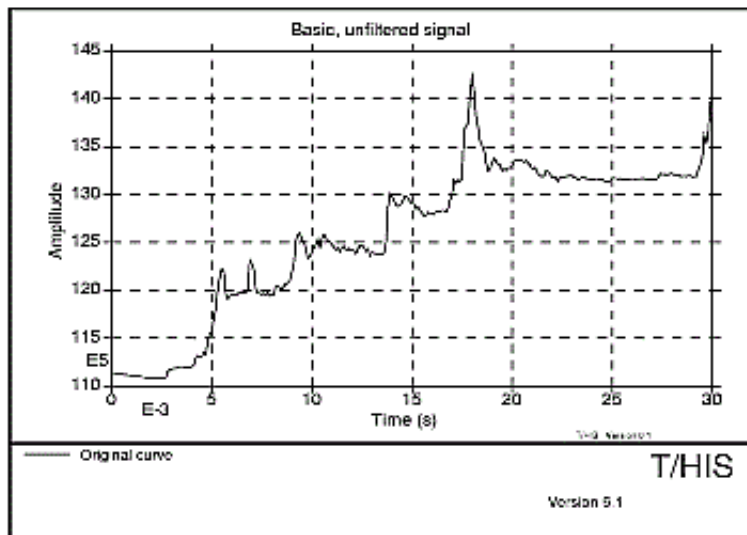
The Butterworth filter is a low pass filter with two input variables; order and cut-off frequency.

The order of the filter controls the roll-off rate, as shown here in the figure (right)

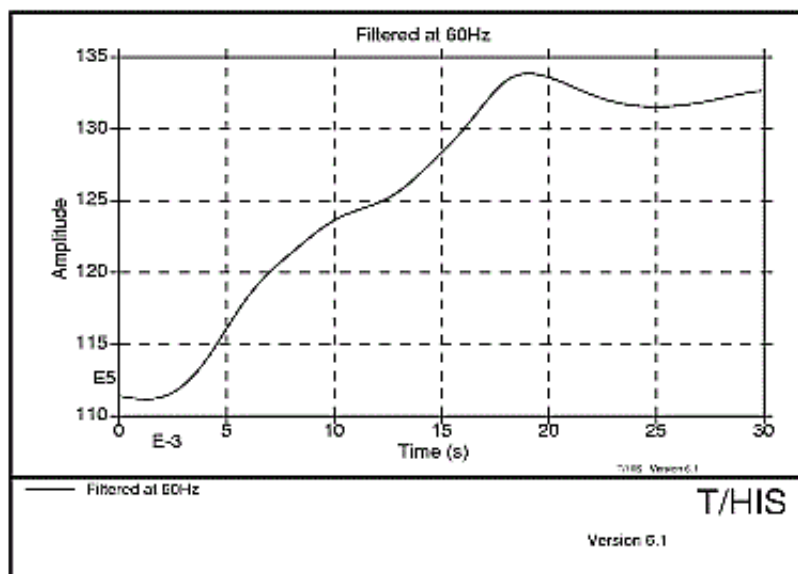
This is a 300Hz filter. It can be seen that higher orders attenuate the results more quickly: they have a higher roll-off rate.



The cut-off frequency is the frequency at which the gain of the filter is -3dB (i.e. the magnitude of signals at this frequency is halved by the filter). The lower the frequency the less noise passes through; but any peaks in the signal tend to get reduced in magnitude and delayed in time.

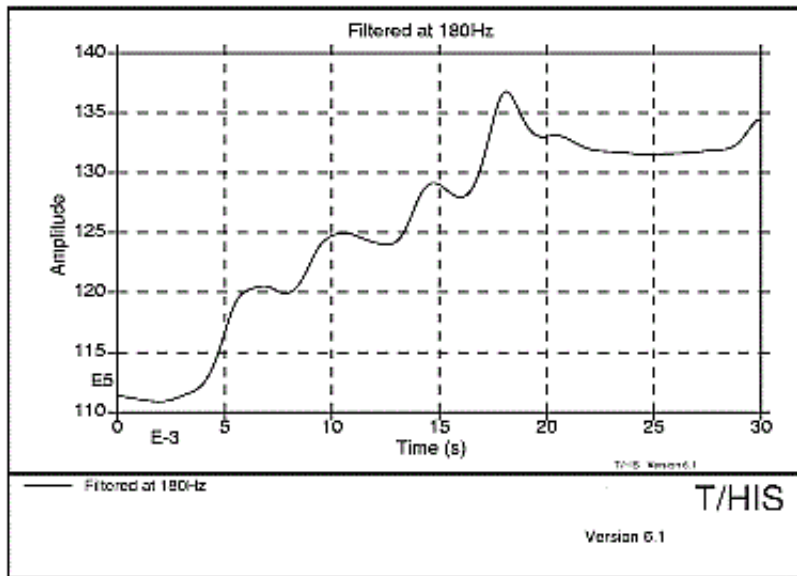


## Unfiltered Signal

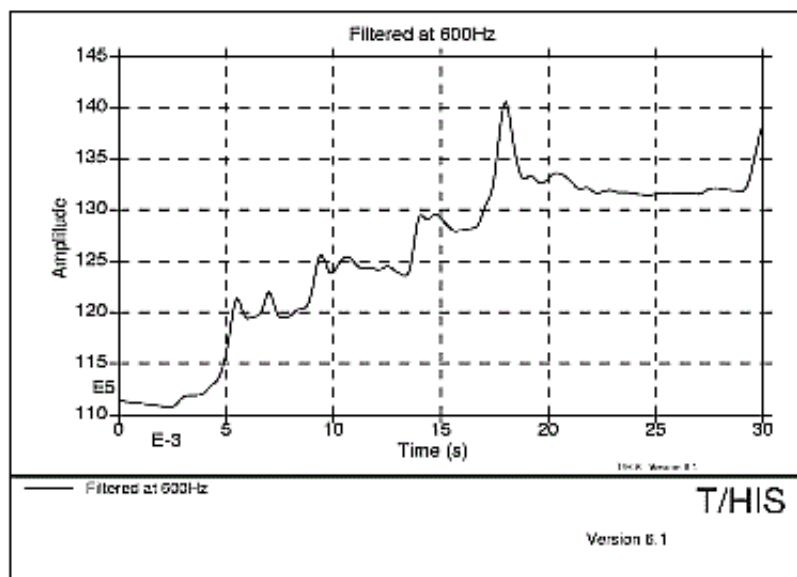


## Filtered at 60Hz

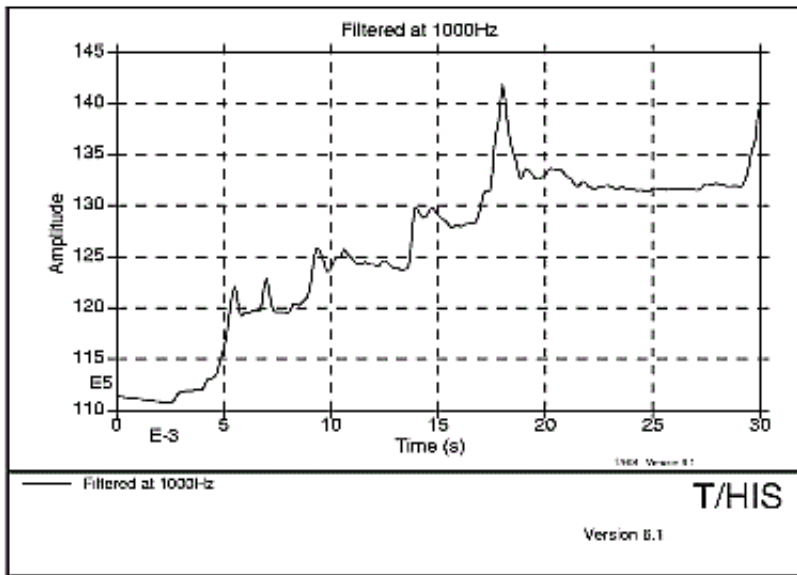




## Filtered at 180Hz



## Filtered at 600Hz



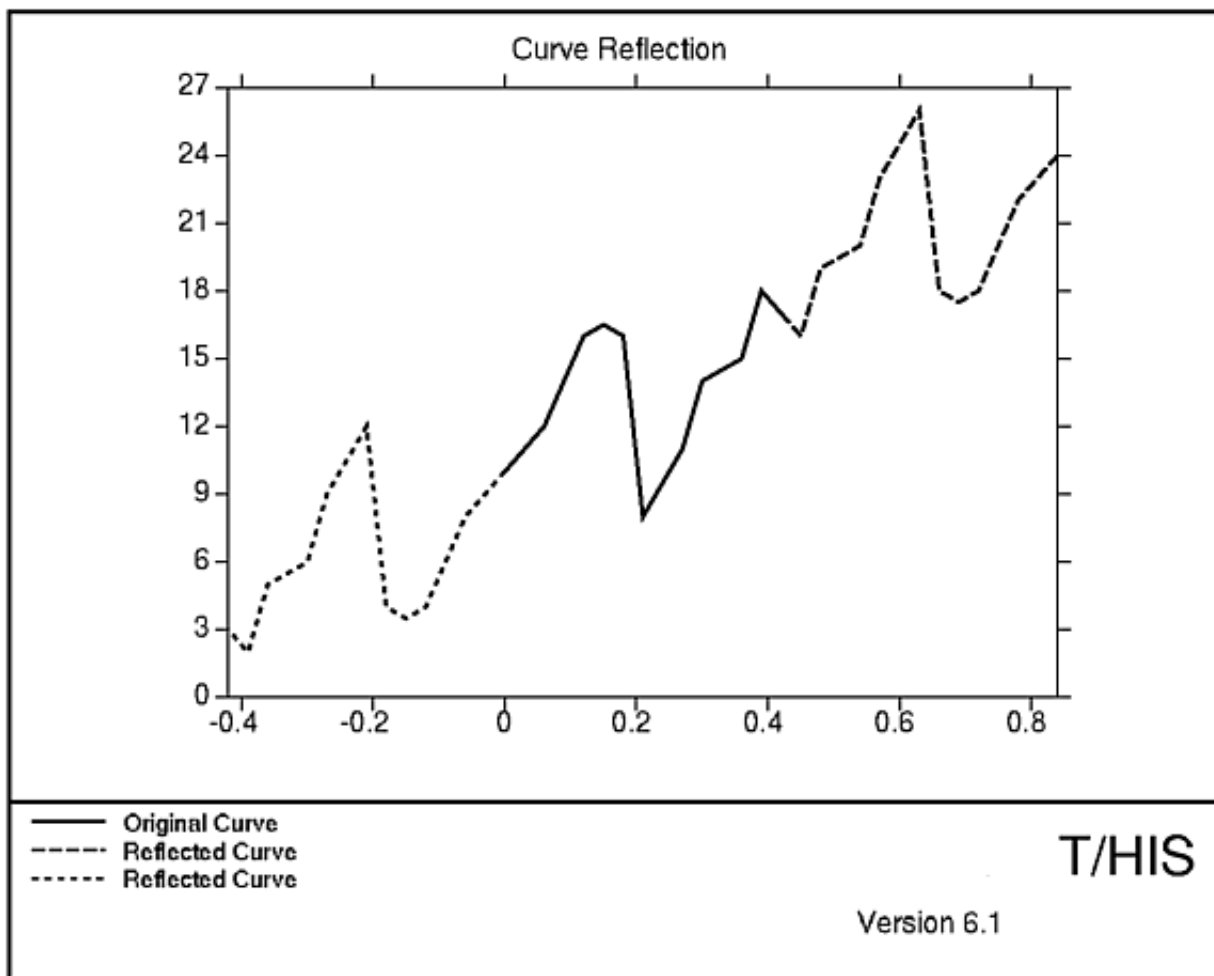
## Filtered at 1000Hz

The above figures show examples of filtering frequency using the four standard SAE filters (60, 180, 600 and 1000 Hz cut-off frequencies: see below). These show clearly how the original signal is smoothed.

### D.3 Butterworth Filter Implementation

Two refinements have been incorporated:

- Reflection of beginning and end of curves to minimise end-effects of filtering (see the figure below).
- The curve is first passed forwards through the filter, then the resulting signal is passed through backwards. This procedure minimises phase change errors. The poles and zeros of the filter are calculated such that the desired cut-off frequency is achieved after two passes.



## D.4 Standard SAE Filter Options

Channel Filter Classes 60, 180, 600 and 1000 are Butterworth filters with the following parameters:

Filter Class	Cut-off Frequency	Order
60	100Hz	2
180	300Hz	2
600	1000Hz	2
1000	1650Hz	2

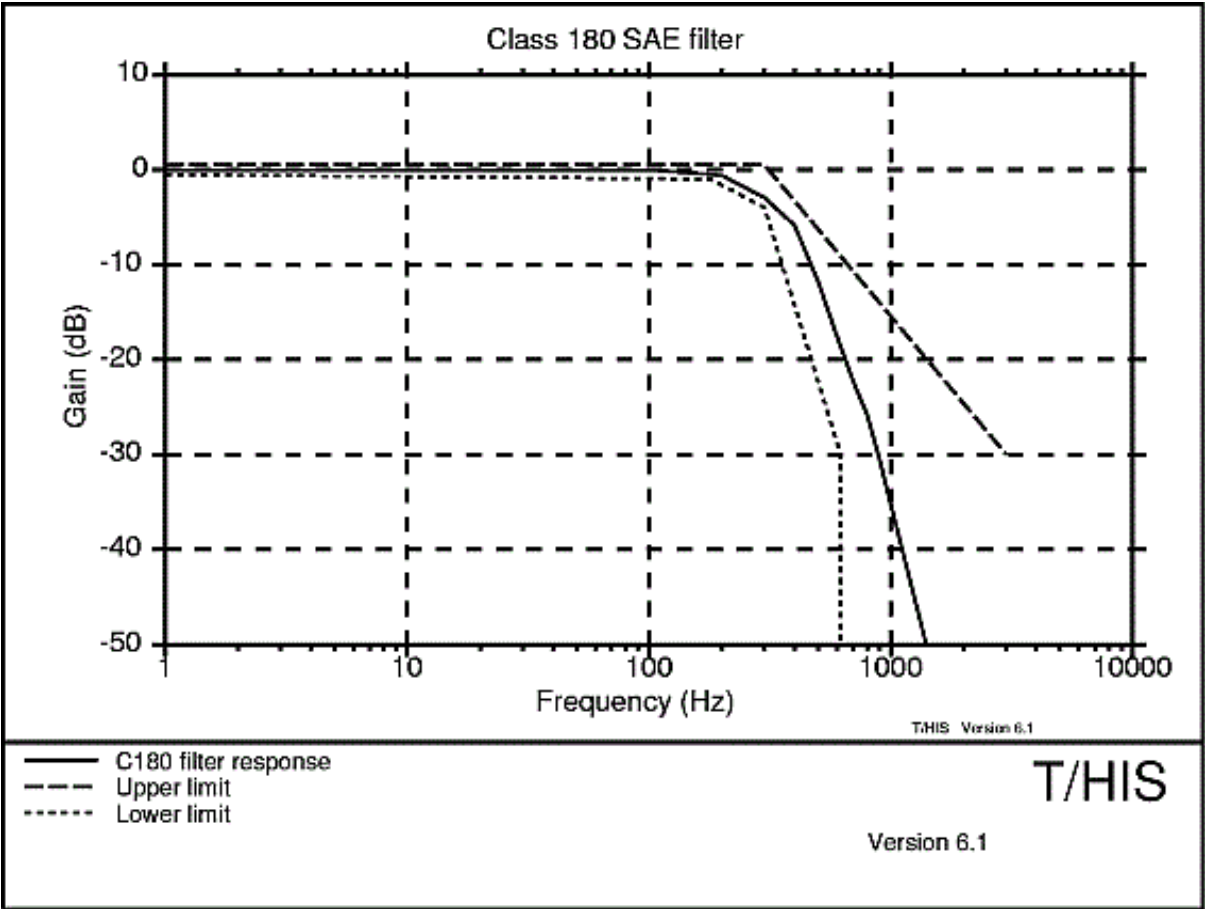
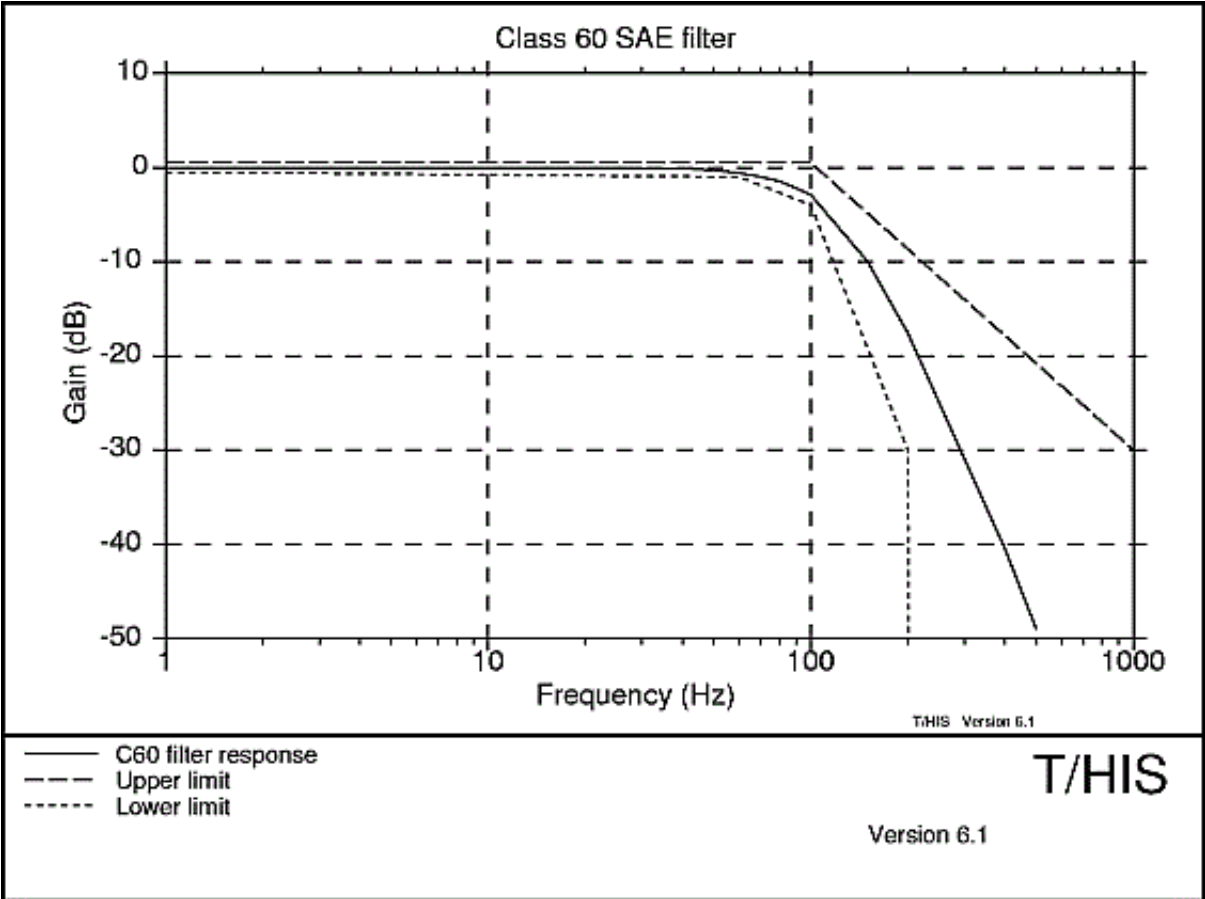
The gain characteristics are compared with the limits given in BS AU228 in the following four figures.

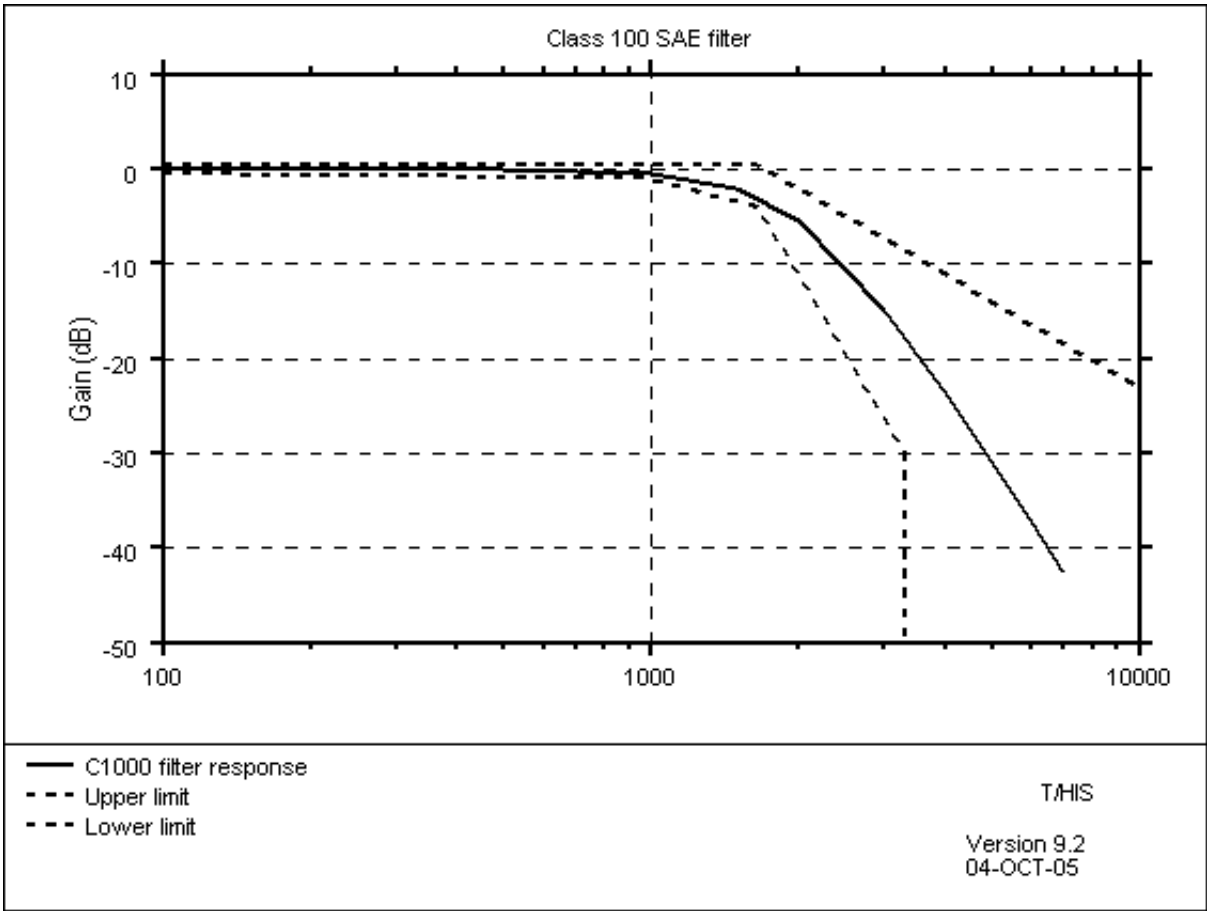
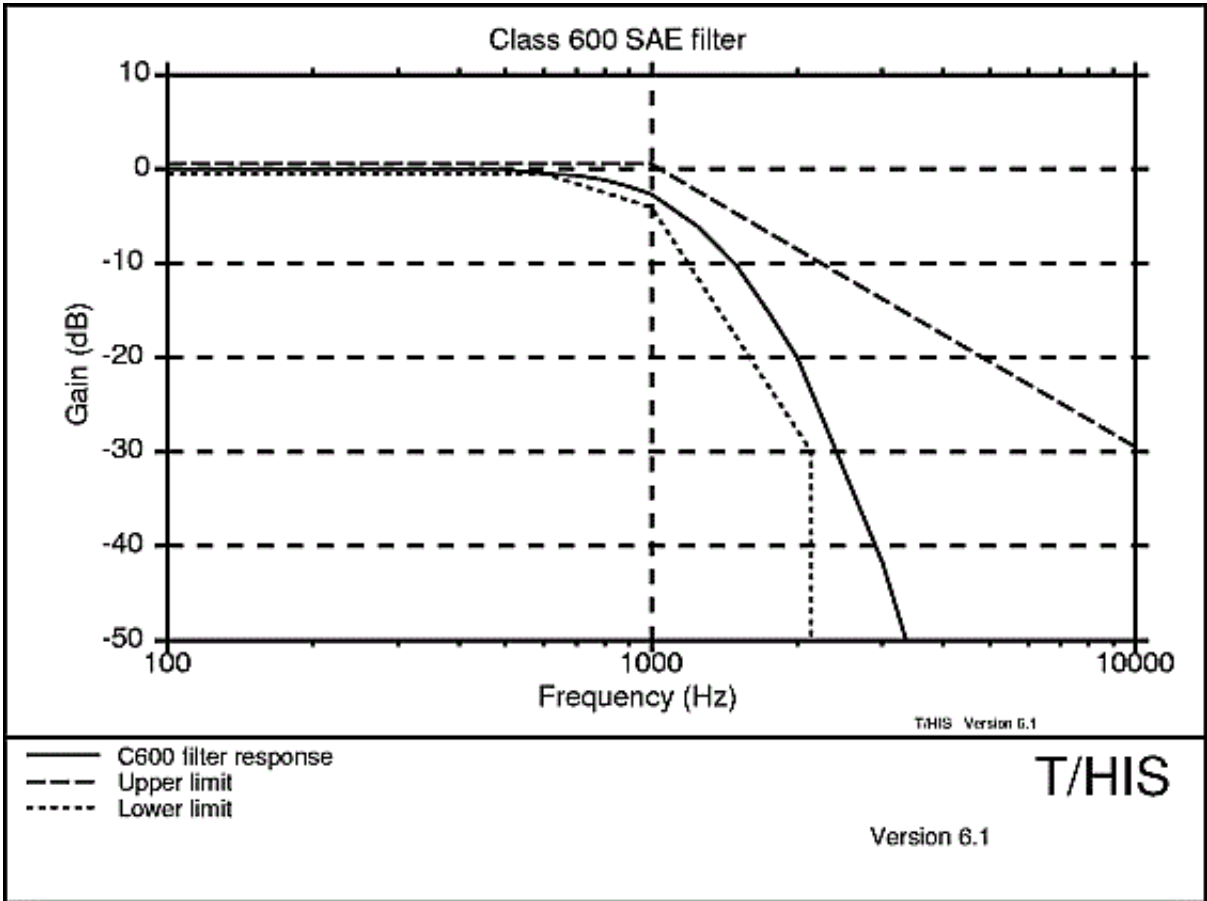
## D.5 Standard FIR filter option

The FIR filter (Finite Impulse Response) is specified by NHTSA. It is used for filtering thoracic accelerations from side impact dummies; the filtered accelerations are then used in calculation of TTI (Thoracic Trauma Index). Its characteristics are:

- A passband frequency of 100Hz.
- A stopband frequency of 189Hz.
- A stopband gain of -50dB.
- A passband ripple of 0.0225dB.

It is based on a standard Fortran programme available from NHTSA.







## APPENDIX E - INJURY CRITERIA

T/HIS has the option to calculate two of the injury criteria that are used currently in occupant protection. These are the head impact criteria or HIC value and 3ms clip value. These criteria are defined as follows:

### E.1 HIC Value

The HIC value is calculated from the resultant acceleration time history of the head centre of gravity filtered through a class 1000 filter. The HIC value is then calculated from;

$$\text{HIC} = \left[ \frac{1}{(t_2 - t_1)} \int_{t_1}^{t_2} a \, dt \right]^{2.5} (t_2 - t_1)$$

Where  $a$  is the acceleration expressed in  $g$ , and  $t_1$  and  $t_2$  are any two points in time. It is now usual for an upper limit on the range  $t_2 - t_1$  of 36ms to be applied.

### E.2 3ms Clip

The 3ms clip value is the maximum value of acceleration that is exceeded for a period of not less than 3 ms. This is not an easily comprehended definition: the following may be of more use:

- At each time point  $T$ , take the interval ( $T$  to  $T+3\text{ms}$ );
- (1) In this interval find the **lowest** acceleration value;
- (2) In this interval find the **lowest** acceleration value;
- (3) The "3ms Clip" value is the interval ( $T$  to  $T+3\text{ms}$ ) which has the **largest** "lowest" value as calculated in (2) above.

So, perhaps, a better definition might be: "the 3ms interval with the highest lowest acceleration value".

### E.3 Viscous Criteria

The VC value is calculated from a compression time history using the following formula;

$$VC = A[V_{(t)}C_{(t)}]$$

where  $C_{(t)} = \frac{D_{(t)}}{B}$

$$V_{(t)} = \frac{8[D_{(t+1)} - D_{(t-1)}] - [D_{(t+2)} - D_{(t-2)}]}{12dt} \quad (\text{ECER95 regulations})$$

$$V_{(t)} = \frac{dD}{dt} \quad (\text{IIHS regulations})$$

$$D_{(t)} = \text{Rib Compression}$$

$$A = \text{Constant (1.3 frontal, 1.0 side)}$$

$$B = \text{Constant (0.229 frontal, 0.140 side)}$$

## E.4 Acceleration Severity Index

The ASI value is calculated from 3 acceleration time histories using the following fomula;

$$ASI_{(t)} = \left[ \left( \frac{ax}{xl} \right)^2 + \left( \frac{ay}{yl} \right)^2 + \left( \frac{az}{zl} \right)^2 \right]^{0.5}$$

Where  $ax, ay, az$  are the X,Y,Z accelerations of the vehicle:

:

- for the 1998 calculation (BS EN 1317-1:1998) they are averaged over a 50ms moving interval.

- for the 2010 calculation (BS EN 1317-1:2010) they are passed through a four-pole phaseless Butterworth filter with a 13Hz cut-off frequency.

$xl, yl, zl$  are acceleration limits  $xl = 12g$   $yl = 9g$   $zl = 10g$ .

The acceleration input curves should be in units of g. If the input curves are in any other unit a conversion factor can be input to convert back to g.

When selecting input curves it is assumed that the X curve is numerically the first curve (the one with the lowest id) of the ones selected and the Z curve is the last. If they are in a different order then the acceleration limits can be modified to reflect the different order. For more information on ASI see BS EN 1317-1.

NOTE: For the BS EN 1317-1:2010 calculation T/His assumes the curves have been filtered through a Class 180 filter and padded with +/-0.5seconds of data as per the specification.

## E.5 Theoretical Head Impact Velocity & Post Impact Head Deceleration

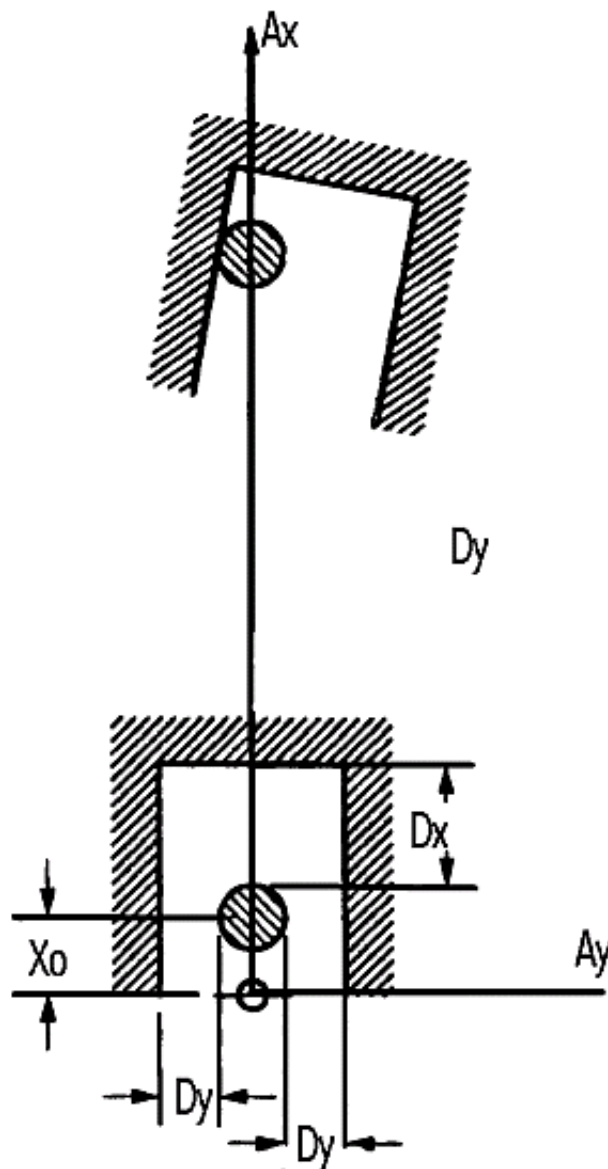


The theoretical head impact velocity concept has been developed for assessing occupant impact severity for vehicles involved in collisions with road vehicle restraint systems. The occupant inside the vehicle is considered to be a freely moving object that, as the vehicle changes its speed due to the contact with the restraint system, continues moving until it strikes the vehicle interior. The velocity magnitude at the time of impact with the vehicle interior is considered to be a measure of the vehicle to vehicle restraint system impact severity.

After impact the head is assumed to continue moving with the vehicle during the rest of the impact event. The post impact head deceleration (PHD) is calculated as the peak value using a 10ms moving average of the resultant vehicle acceleration after the THIV impact.

The THIV calculation requires the following inputs

- Horizontal Vehicle Acceleration Time History ( $A_x$ )
- Lateral Vehicle Acceleration Time History ( $A_y$ )
- Yaw Rate Time History
- Horizontal Distance from the occupants head to vehicle ( $D_x$ )
- Lateral Distance from the occupants head to vehicle ( $D_y$ )



For more information on THIV and PHD see BS EN 1317-1.

### E.6 Biomechanical neck injury predictor (NIJ)

The biomechanical neck injury predictor is a measure of the injury due to the load transferred through the occipital

condyles. Its calculation combines the neck axial force and the flexion/extension moment about the occipital condyles.

It is used in association with the USSID dummy for standard American frontal impact tests.

The shear force ( $F_x$ ), axial force ( $F_z$ ) and bending moment ( $M_y$ ) are measured by the dummy upper neck load cell for the duration of the crash, using force and moment definitions consistent with SAE J221/1. T/HIS will calculate the bending moment using the equation:

$$M_y = M_y' - e \cdot F_x$$

Where  $e$  is the  $e$  distance specified in the input window,  $F_x$  is the shear force.

Shear force, axial force and bending moment must be filtered using an SAE Channel Frequency Class 600 filter (C600) for the purposes of calculation.

During the collision, the Axial Force ( $F_z$ ) can be in either tension or compression whilst the occipital condyle bending moment ( $M_{ocy}$ ) can be in either flexion or extension. This results in 4 possible loading conditions corresponding to the 4 curves output by T/HIS; tension-extension (Nte), tension-flexion (Ntf), compression-extension (Nce), and compression-flexion (Ncf). At each point in time only one of these 4 conditions can be met, hence the NIJ value is calculated for that condition and the value for the other 3 conditions is considered a value of zero..

The expression for calculating each NIJ loading condition is given by:

$$NIJ = (F_z/F_{zc}) + (M_{ocy}/M_{yc})$$

where  $F_z$  and  $M_{ocy}$  are as defined above,  $F_{zc}$  and  $M_{yc}$  refer to the axial force and Bending moment critical values, given below:

The values of  $F_{zc}$  and  $M_{yc}$  vary depending on the occupant, the occupants position and the sign of  $F_z$  and  $M_{ocy}$

For the dummy to pass the test, the following conditions must be met:

- (i) None of the 4 NIJ values may exceed 1.0 at any time during the event
- (ii) Peak Tension Force ( $F_z$ ), measured at the upper neck load cell, may not exceed the specific dummy's limit (e.g. 2070N for the Hybrid III small female) at any time
- (iii) Peak Compression Force ( $F_z$ ), measured at the upper neck load cell, may not exceed the specific dummy's limit (e.g. 2520N for the Hybrid III small female) at any time

For more information on the use and calculation of NIJ, refer to the FMVSS 208 document

## E.7 The Thoracic Trauma Index (TTI)

The Thoracic Trauma Index is used as a predictor of thoracic injury severity in the USSID dummy in standard American Side Impact tests.

The Index considers both rib and Thorax acceleration in an impact.

The expression for calculating TTI is given by:

$$TTI = (G(R) + G(LS))/2$$

Where  $G(R)$  is the greater of the peak accelerations of either the upper or lower rib, expressed in g, and  $G(LS)$  is the peak acceleration in the lower spine (T12), expressed in g.

For the dummy to pass the test, the following conditions must be met:

- (i) The TTI value must not exceed;
  - (a) 85g for a passenger car with 4 side doors, and for any multipurpose vehicle, truck or bus
  - (b) 90g for a passenger car with 2 side doors
- (ii) The peak lateral acceleration of the pelvis shall not exceed 130g
- (iii) Any side door, struck by the moving deformable barrier, shall not separate totally from the car.
- (iv) Any door not struck by the moving deformable barrier must meet the following requirements;
  - (a) The door shall not disengage from the latched position
  - (b) The latch shall not separate from the striker

- 
- (c) The hinge components shall not separate from each other or from their attachment to the vehicle
  - (d) Neither the latch nor the hinge systems of the door shall pull out of their anchorage

For more information on the use and calculation of TTI, refer to the FMVSS 214 document



## APPENDIX F - Curve Correlation

### COR1 and COR2

The Correlation functions COR1 and COR2 provide a measure of the degree to which two curves match. When comparing curves by eye, the quality of correlation may be judged on the basis of how well matched are the patterns of peaks, the overall shapes of the curves, etc, and can allow for differences of timing as well as magnitude. Thus a simple function based on the difference of Y-values (such as T/HIS ERR function) does not measure correlation in the same way as the human eye. The T/HIS correlation function attempts to include and quantify the more subtle ways in which the correlation of two curves may be judged.

The correlation function may be applied to any two curves whose x-values increase monotonically (e.g. responses versus time). The results are independent of the units used, e.g. milliseconds or seconds are both acceptable. The sign of the y-values is not important.

Only the overlap time period is considered (i.e. the range of x-values for which both curves have a y-value). The time period (range of X-values) and maximum absolute Y-value are used to non-dimensionalise the curves such that x-values run from 0 to 1, and the maximum absolute y-value is 1.

Five measures of correlation are calculated. Each is given equal weighting. The final correlation score is given as a percentage - two identical curves would score 100%.

The first two measures require identification of peaks in the curves. An unlimited number of peaks in each curve will be considered. A peak is defined as a local maximum (or in the case of negative y-values a minimum), satisfying the following criteria:

- Absolute y-value at least 0.5
- Separated from any larger peak by a trough (local minimum) at least 0.2 deep.

Peaks of positive or negative signs are considered. Peaks are matched only against peaks of the same sign in the other curve.

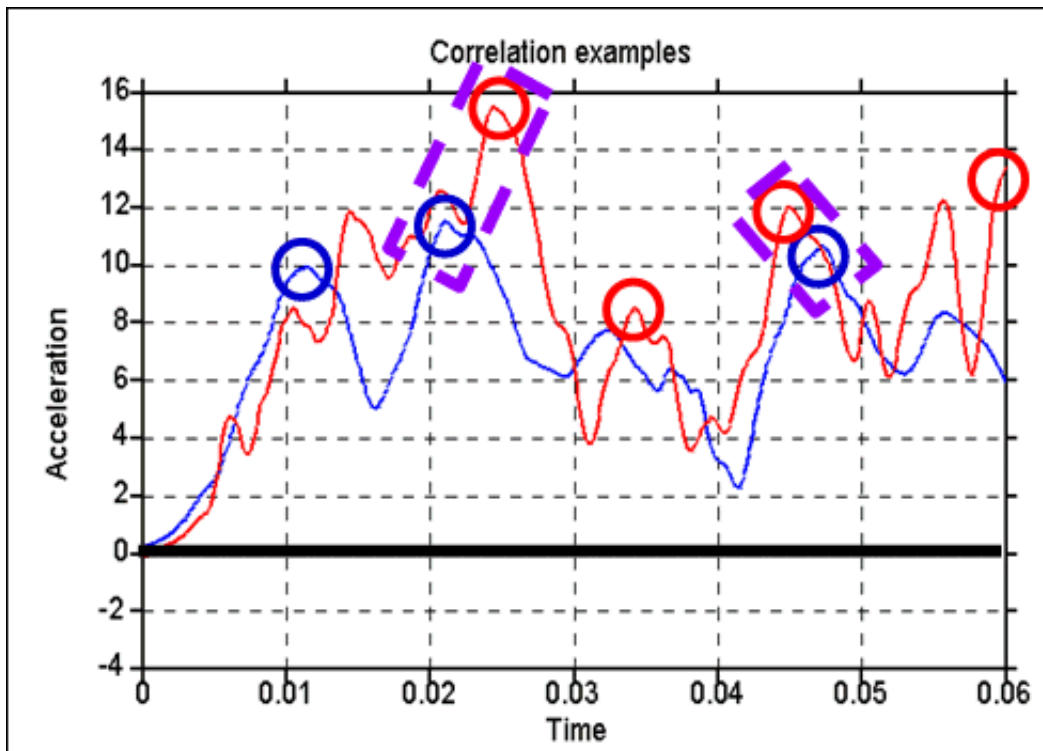
#### Measure 1 - Peak values

For each identified peak in Curve A, find the maximum value in Curve B within the same time range for which the value in Curve A is within a tolerance of the peak value. Points are lost according to the error in y-values compared to a tolerance limit. Repeat for peaks in curve B against values in Curve A.

This measure allows for the situation where curves are similar but the peaks are more strongly delineated in one of the curves, such that the program does not recognise the other curve as having a peak in that location.

#### Measure 2 - Peak matching

For each identified peak in Curve A, find the closest identified peak in Curve B. Points are lost according to the largest error (timing or y-value) compared to tolerance limits; points are also lost if there is no corresponding peak in Curve B. Repeat for Curve A peaks matched against those of Curve B.



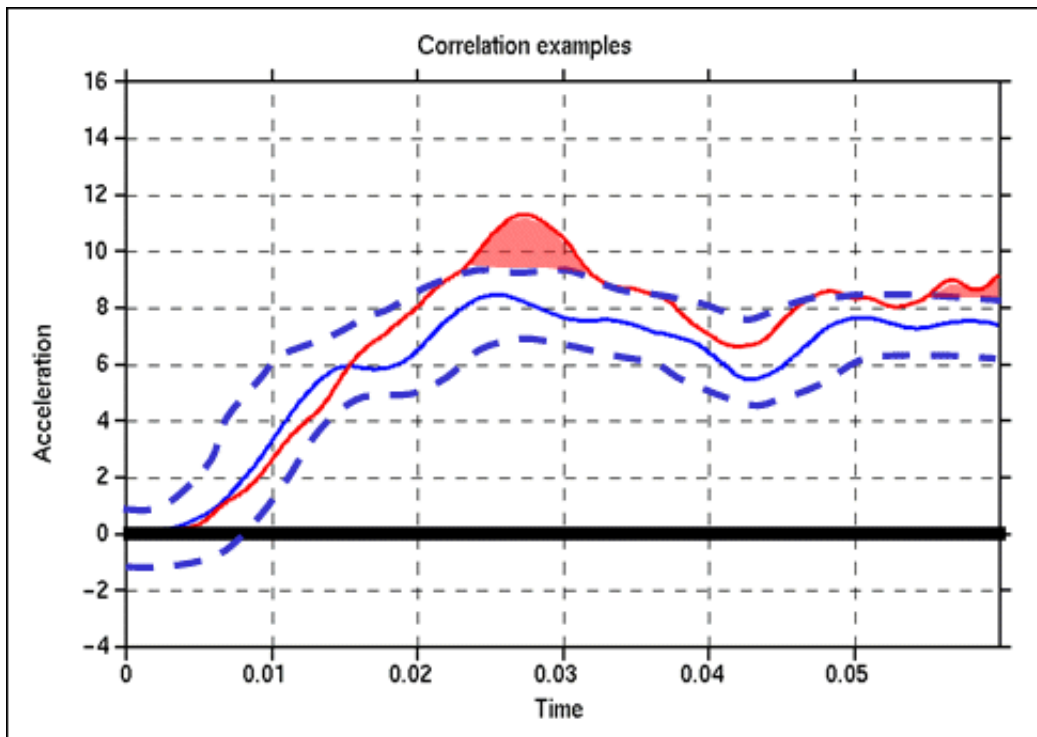
This measure picks up matching of primary and secondary peaks in the curves, which may correspond to physical events.

### Measure 3 - Area matching

The integral of each curve is calculated by summing the area of the curve above  $y=0$  and the absolute area of the curve below  $y=0$ . Points are lost according to the difference compared to a tolerance limit.

### Measure 4 - Curve shape (low frequency excursion)

The curves are filtered. A band is drawn around filtered curve A (using positive and negative offsets in  $x$  and  $y$ ). The area of excursions of filtered Curve B outside the band is calculated. Points are lost according to the excursion area compared to a tolerance limit. The process is repeated for filtered Curve A excursions from a band drawn around filtered curve B



### Measure 5 - Curve shape (full curve)

The same as Measure 4 except that the curves are not filtered and different tolerance limits and band sizes may be used.

### Output

T/HIS prints the overall correlation percentage and the marks from each measure to the screen or to a text file. A new curve is created from each input curve showing the identified peaks (used in measures 1 and 2). As the same curve could be used as input to multiple correlations the correlation percentage is stored internally in T/HIS with the 2 output curves NOT the input curves.

The correlation percentage can be accessed from within FAST-TCF scripts by requesting the "correlate" property for either of the 2 output curves.

**e.g. operation correlate strict curve\_1 curve\_2 tag curve\_3 curve\_4**

Calculate correlation between "curve\_1" and "curve\_2". Tag the curves containing the peaks as "curve\_3" and "curve\_4"

**tab output.txt curve\_3 correlate**

Output the curve correlation value from "curve\_3" to the file "output.txt"

**taba output.txt curve\_4 correlate**

Append the curve correlation value from "curve\_4" to the file "output.txt"

### Selection of Parameters

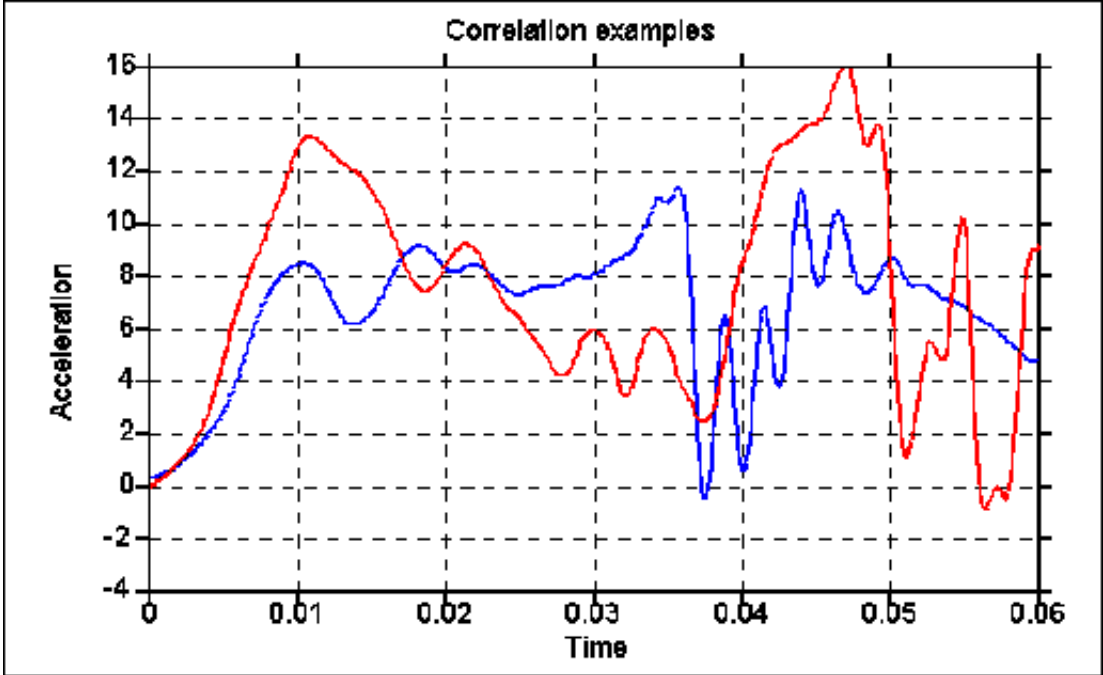
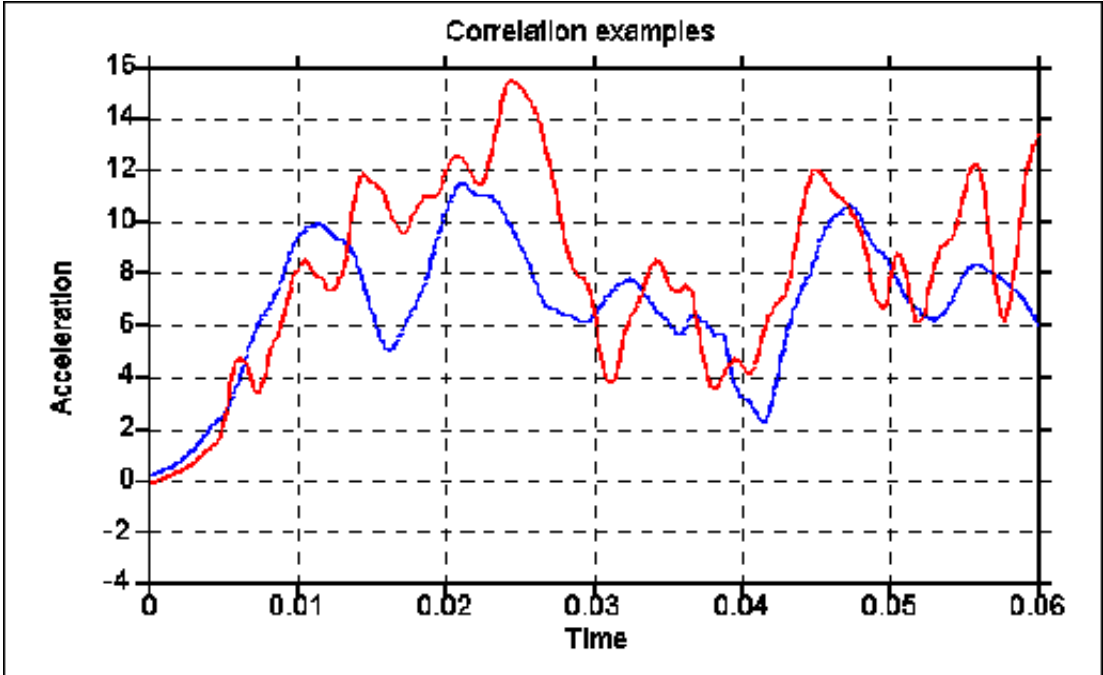
The Correlation algorithm has many tolerance limits and other inputs. Two sets of these parameters have been pre-selected, to offer strict or less strict judgement of correlation (buttons COR1 and COR2 in the Automotive menu). The parameters selected are:

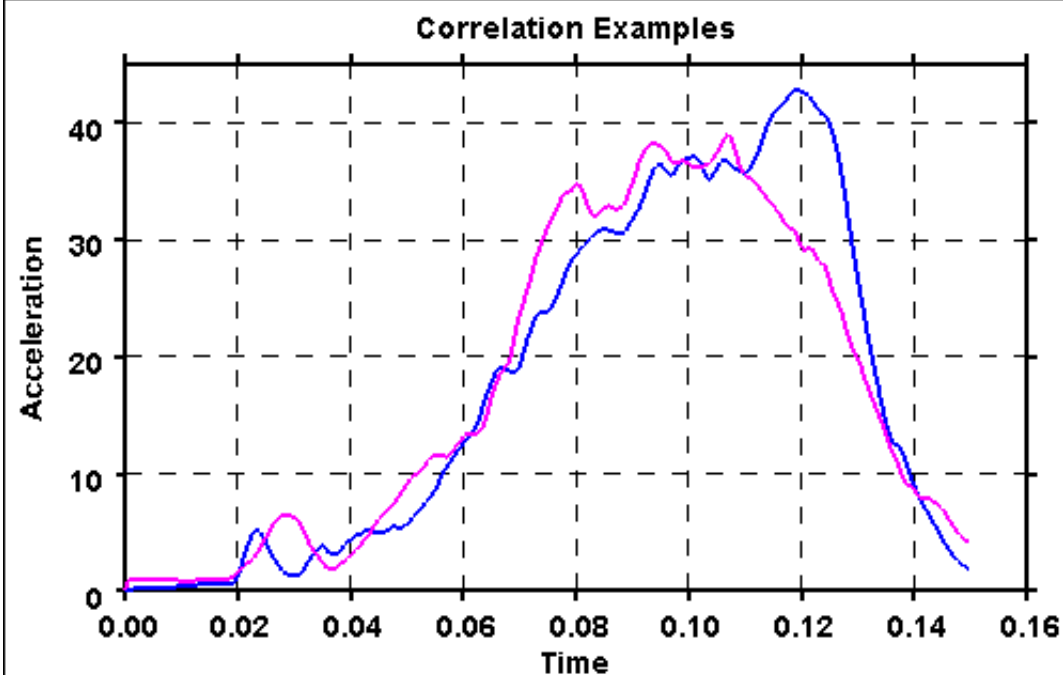
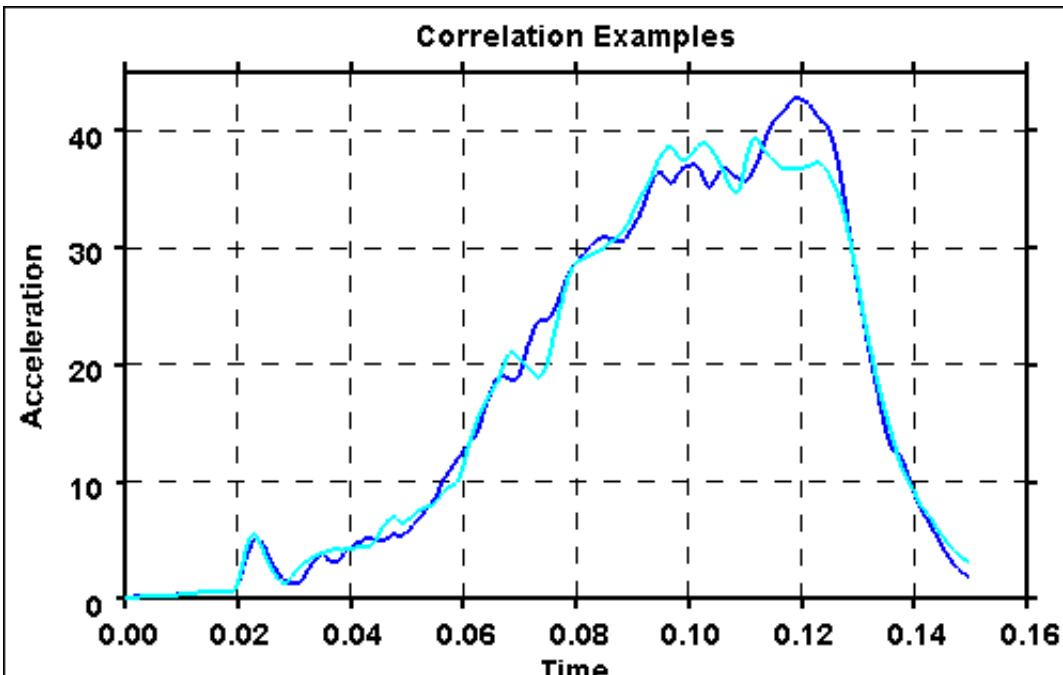
Criterion	Description	COR1 Value	COR2 Value
Peak matching	Fraction difference in timing that scores zero points for this peak	0.2	0.4
Peak matching and peak values	Fraction difference in value that scores zero points for this peak	0.25	0.5

Area matching	Fraction difference in integral that scores zero points	0.3	0.5
Curve shape (low frequency trend)	Size of tolerance band in X and Y, as fractions of the curve extent in X and Y	0.025	0.05
Curve shape (low frequency trend)	Excursion area fraction scoring zero points	0.1	0.2
Curve shape (full curve)	Size of tolerance band in X and Y, as fractions of the curve extent in X and Y	0.025	0.05
Curve shape (full curve)	Excursion area fraction scoring zero points	0.2	0.4

It is expected that, if COR1 rates Curves A and B as better correlated than C and D, then COR2 would also rate the pairs of curves in the same order. The percentage correlation would be greater in each case from COR2 than from COR1. COR1 will provide a greater difference (discrimination) between well-correlated and very well-correlated pairs of curves; while COR2 will provide greater discrimination between averagely-correlated and poorly-correlated pairs of curves. The purpose of offering both versions of the correlation function is to allow the user to select a calibration of the function appropriate to the typical input curves used.

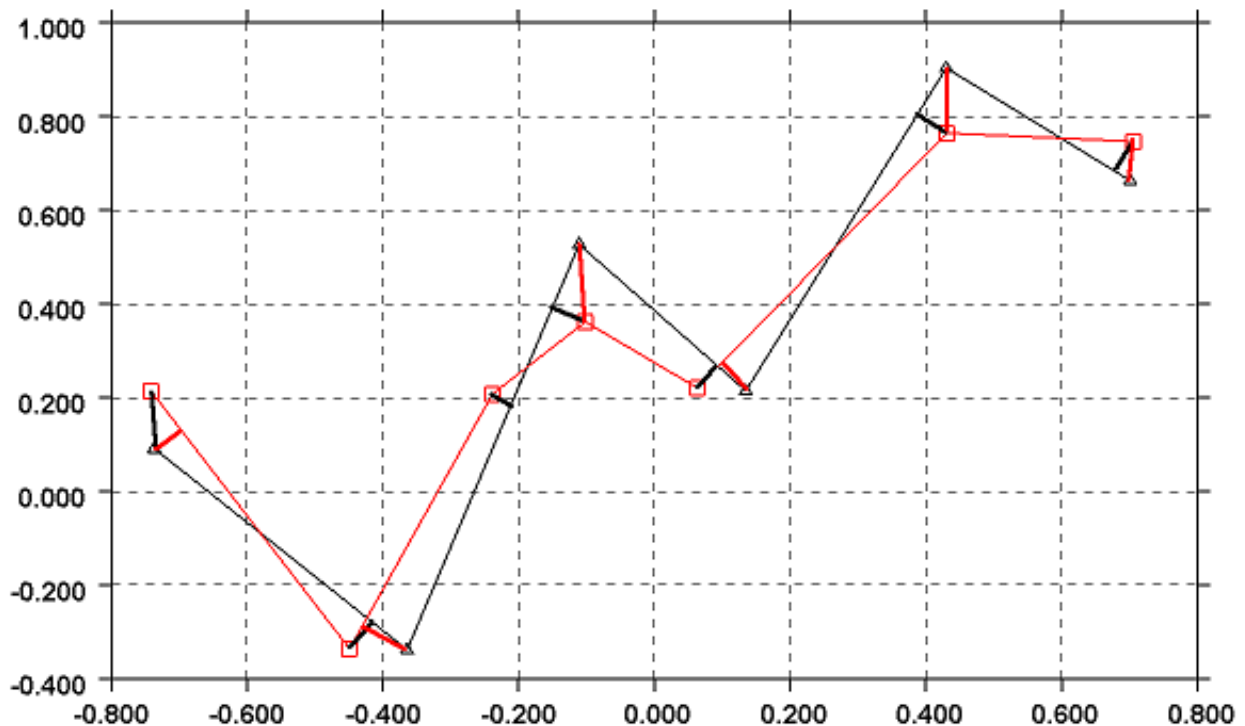


Examples	COR1	COR2
<div><p>Correlation examples</p></div>	17%	42%
<div><p>Correlation examples</p></div>	27%	62%

<div><p>Correlation Examples</p></div>	70%	88%
<div><p>Correlation Examples</p></div>	84%	92%

### COR3

The Correlation function COR3 provides another measure of the degree to which two curves match based on the distance between the two curves.



This function first normalises the curves using two factors, specified either by the user or defaults calculated by the program (the maximum absolute X and Y values of both graphs).

For each point on the first normalised curve, the shortest distance to the second normalised curve is calculated (the thick black lines on the image above). The root mean square value of all these distances is subtracted from 1 and then multiplied by 100 to get an index between 0 and 100.

The process is repeated along the second curve (the thick red lines show the distances) and the two indices are averaged to get a final index. The higher the index the closer the correlation between the two curves.

Note that the choice of normalising factors is important. Incorrect factors may lead to a correlation index outside the range of 0 to 100.

### WIF

The Correlation function WIF provides another measure of the degree to which two curves match. It uses the Weighted Integrated Factor method:

$$crit = 1 - \sqrt{\frac{\sum \max(f[n]^2, g[n]^2) \cdot \left(1 - \frac{\max(0, f[n] \cdot g[n])}{\max(f[n]^2, g[n]^2)}\right)^2}{\sum \max(f[n]^2, g[n]^2)}}$$



## APPENDIX G - The ERROR Calculation

The ERROR function outputs a number of values to indicate the degree of correlation between 2 curves. The function requires two input curves

- A reference curve to compare against ( the first curve selected )
- The curve to compare against the reference

Once 2 curves have been selected the a check is carries out to see if the two curves contain the same number of points and if the range of x-axis values the same for the two curves. If any inconsistencies are found then a warning message is generated.

The following values are then calculated

Maximum difference and time of variation

Maximum difference as a %age of the reference value at the same time

Maximum difference as a %age of the peak reference value

Average difference

Average difference as a %age of the peak reference value

$$\text{Area weighted difference} = \left( \frac{\int |y_r - y_c| dx}{\frac{1}{2} \left( \int y_r dx + \int y_c dx \right)} \right)$$

where  $y_r$  = Reference Curve  
 $y_c$  = Data Curve

T/HIS Regression coefficient.

$$R^2 = \left[ 1 - \frac{\sum (y_c - y_a)^2}{\sum y_a^2 - \frac{\sum y_a^2}{n}} \right]$$

$y_c$  = Data Curve

$y_a$  = Average of Data and Reference Curve =  $\frac{1}{2}(y_c + y_r)$

$n$  = Number of Data Points

This is a value between 0 and 1 where 1 means 100% correlation



## APPENDIX H - The "oa\_pref" preference file

This file contains code-specific preferences that can be used to modify the behaviour of T/HIS. It is optional and, where entries (or the whole file) are omitted T/HIS will revert to its default settings.

### "oa\_pref" naming convention and locations

The file is called "oa\_pref". It is looked for in the following places in the order given:

- The optional administration directory defined by the environmental variable (`$OA_ADMIN` or `$OA_ADMIN_xx` where xx is the release number).
- The site-wide installation directory defined by the environment variable (`$OA_INSTALL`)
- The user's home directory: `$HOME` (Unix/Linux) or `%USERPROFILE%` (Windows)
- The current working directory

See [Installation organisation](#) for an explanation of the directory structure.

All four files are read (if they exist) and the last preference read will be the one used, so the file can be customised for a particular job or user at will.

Files do not have to exist in any of these locations, and if none exists the programme defaults will be used.

### On Unix and Linux:

`$HOME` on Unix and Linux is usually the home directory specified for each user in the system password file. The shell command "`printenv`" (or on some systems "`setenv`") will show the value of this variable if set. If not set then it is defined as the "~" directory for the user. The command "`cd; pwd`" will show this.

### On Windows:

`%USERPROFILE%` on Windows is usually `C:\Documents and Settings\<user id>\`. Issuing the "`set`" command from an MS-DOS prompt will show the value of this and other variables.

Generally speaking you should put

- Organisation-wide options in the version in `$OA_ADMIN_xx` and/or `$OA_INSTALL`,
- User-specific options in `$HOME` / `%USERPROFILE%`
- Project-specific options in the current working directory.

The file contains preferences for the SHELL (lines commencing `shell*`), THIS (lines commencing `this*`), D3PLOT (lines commencing `d3plot*`), PRIMER (lines commencing `primer*`) and REPORTER (lines commencing `reporter*`). All lines take the format `<preference name> <preference value>`.

The general copy of the preference file should be present in the [\\$OA\\_ADMIN\\_xx](#) and/or [\\$OA\\_INSTALL](#) directory. This should contain the preferences most suitable for all software users on the system.

An individual's specific preferences file can be stored in the individual's home area. This can be used to personally customise the software to the individual's needs.

Whenever one of the programs whose preferences can be stored in the `oa_pref` file is fired up, the program will take preferences first from the general preference file in the [\\$OA\\_ADMIN\\_xx](#) directory (if it exists) then the [\\$OA\\_INSTALL](#) directory, then from the file in the user's home area, then from the current working directory.

Preferences defined in the general `oa_pref` file can be modified in the user's personal file but they can't be removed by it.

From version 9.4 onwards preferences can be locked. If a preference is locked it cannot be changed in an `oa_pref` file in a more junior directory. To lock a preference use the syntax `'this#'` rather than `'this*'`.

## The interactive Preferences Editor

You are free to edit `oa_pref` files by hand, but there is an interactive "Preferences Editor" that may be called from within T/HIS that makes the job much easier.

It is started by [Options, Edit Preferences](#) or through the Preferences Button in the Tool menu

The preferences editor reads an XML file that contains all possible preferences and their valid options, and allows you to change them at will. In this example the user is changing the background colour in D3PLOT.

Note that changes made in the Preferences editor will not affect the current session of D3PLOT, they will only take effect the next time it is run.

If you have write permission on the oa\_pref file in the \$OASYS directory you will be asked if you want to update that file, otherwise you will only be given the option of updating your own file in your \$HOME / \$USERPROFILE directory.

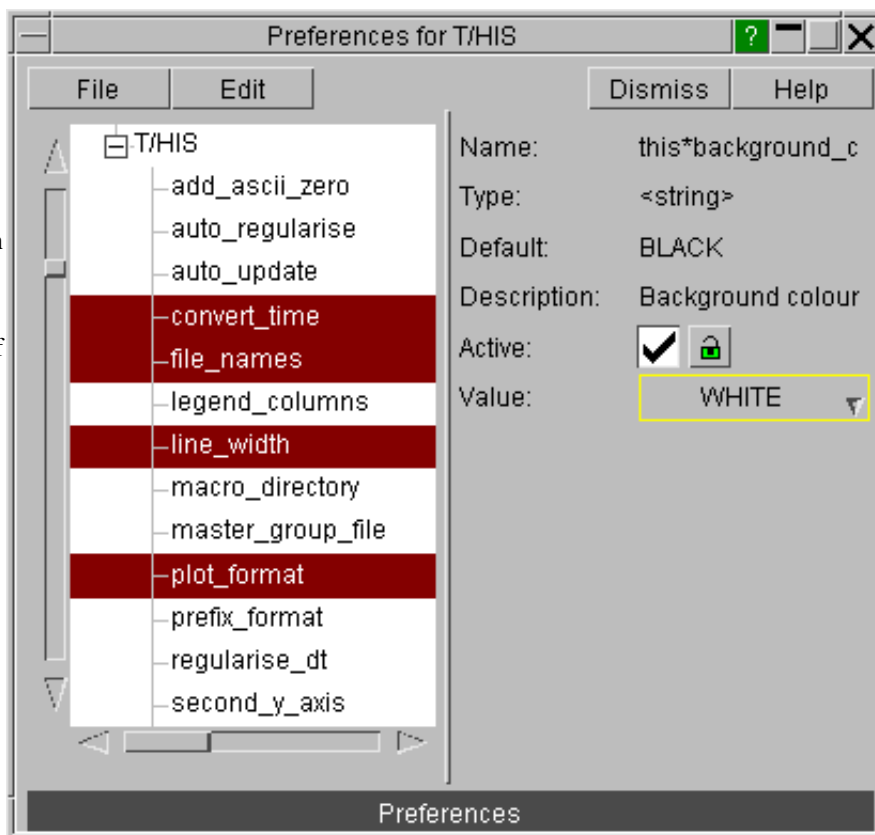
In this example the user is changing the background colour.

The option is "active" (ie present in the oa\_pref file) and currently is set to WHITE.

Usage is:

- Select an option in the Tree on the left hand side
- Make it active / inactive
- If active select a value from the popup, or type in a value if necessary

The colour of the highlighting in the left hand side tree is significant:



Green

Means that the option has been read from your \$HOME/\$USERPROFILE file.

Red

Means that the option has been read from the \$OA\_INSTALL file.

Magenta

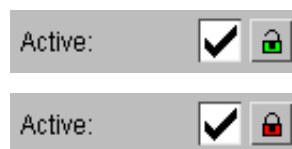
Means that the option had been read from the \$OA\_ADMIN file.

In either event, regardless of the data source, the updated option will be written to the file chosen when you started the preferences editor.

Because of the order of file reading ([see above](#)), and option read from the master \$OASYS file, amended, and written to your local \$HOME file will take precedence when you next run T/HIS.

## Locking Preferences

From version 9.4 onwards preferences can be locked. Beside each option in the preference editor is a padlock symbol. If the symbol is green then the option is unlocked, if it is red then it is locked. If a preference option has been locked in a file that the user can not modify then an error message will be generated if the user tries to edit that option.



If a user manually edits the "oa\_pref" file to try and set an option that has been locked in another preference file then the option will be ignored in the users preference file.



## Format of the `oa_pref` file

Entries are formatted in the following way: `<programme>*<option>: <setting>`

For example: `this*laser_paper_size: A4`

The rules for formatting are:

- The `<programme>*<option>` string must start at column 1;
- This string must be in lower case, and must not have any spaces in it.
- The `<setting>` must be separated from the string by at least one space.
- Lines starting with a "#" are treated as comments and are ignored.

(Users accustomed to setting the attributes of their window manager with the `.Xdefaults` file will recognise this format and syntax.)

### "oa\_pref" arguments valid for T/HIS.

Preference	Type	Description	Valid arguments	Default
add_ascii_zero	<logical>	Automatically add point at time zero if required	TRUE, FALSE	FALSE
auto_regularise	<logical>	Always regularise curves before filtering	TRUE, FALSE	FALSE
auto_update	<logical>	Automatically replot graph after changing axis/title options	TRUE, FALSE	TRUE
checkpoint_dir	<string>	Directory for checkpoint files, or "none" to suppress them altogether		<none>
error_handler	<string>	how to handle errors and exceptions	no_action, mini_dump, trap_continue	mini_dump
convert_time	<logical>	Automatically convert from ms->s when filtering	TRUE, FALSE	FALSE
file_names	<string>	Controls default file filters. LSTC = d3thdt*, xtf*, OASYS/ARUP = *.thf, *.xtf	OASYS, ARUP, LSTC	OASYS
legend_columns	<string>	Number of columns to display in legend	1, 2, 3	2
line_width	<real>	Default line width for curves (pixels)	1.0, 2.0, 4.0, 8.0	2.0
datum_file	<string>	File containing DATUM line definitions		<none>
macro_directory	<string>	Specify a directory for T/HIS to look in for MACRO definitions		\$OA_INSTALL/ this_library/macros
master_group_file	<string>	Filename for default group information		<none>

read_group_files	<string>	Default action when a group file is found in a model directory and T/HIS has already read a group file.	IGNORE, DELETE, OVERWRITE, INCREMENT	IGNORE
plot_format	<string>	Default format of plot	COLUMN, DEFAULT, AUTO, OFF, FULL, FLOATING	COLUMN
prefix_format	<string>	Prefix for	MODEL, DIRECTORY, ROOTNAME, USER	MODEL
regularise_dt	<real>	Time interval for automatic curve regularisation		0.0001
second_y_axis	<logical>	Display 2nd y axis	TRUE, FALSE	FALSE
show_hic_value	<string>	Display HIC value	ON, OFF	OFF
show_3ms_value	<string>	Display 3ms Clip value	ON, OFF	OFF
show_thiv_value	<string>	Display THIV value	ON, OFF	OFF
show_phd_value	<string>	Display PHD value	ON, OFF	OFF
injury_text_colour	<string>	Colour used to display injury criteria values	FOREGROUND, CURVE, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
injury_line_colour	<string>	Colour used to display injury criteria lines	FOREGROUND, CURVE, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
auto_blank	<string>	Turn ON/OFF AutoBlank	ON, OFF	ON
auto_blank_mode	<string>	Set the default AutoBlank mode	MODEL, COMPONENT_ID, ENTITY_TYPE, ENTITY_ID, COMPONENT_TYPE, SURFACE, CURVE	MODEL
start_in	<string>	Directory to start T/HIS in		<none>
vc_method	<string>	Default method for calculating Viscous Criteria	ECER95, IIHS	ECER95
asi_method	<string>	Default method for calculating Acceleration Severity Index	2010, 1998	2010

curve_palette	<string>	Controls how many colours are used by curves, default(6), extended(13), no_grey(27), full(30+any user defined)	DEFAULT, EXTENDED, NO_GREY, FULL	OFF
ftcf_error_count	<integer>	Maximum number of errors before a FAST-TCF script terminates		10
null_value	<real>	Value to assign to curves when data doesn't exist		1.0E+18
csv_separator	<string>	CSV file field separator	COMMA, TAB, SPACE	COMMA

The following options control the automatic creation of curve groups.

Preference	Type	Description	Valid arguments	Default
group_by_model	<logical>	Automatically create a curve group for each model	TRUE, FALSE	TRUE
group_by_type	<logical>	Automatically create a curve group for each entity type data is read for	TRUE, FALSE	FALSE
group_by_component	<logical>	Automatically create a curve group for each component type data is read for	TRUE, FALSE	FALSE
component_group_name	<string>	Controls how curve groups created for components are named)	COMPONENT, COMPONENT_AND_TYPE	COMPONENT
group_by_file_index	<logical>	Automatically create a curve group based on the index of a curve read from a curve file	TRUE, FALSE	FALSE

The following options control the columns that are displayed by default in the curve table

Preference	Type	Description	Valid arguments	Default
ctable_show_curve_id	<logical>	Display Curve IDs	TRUE, FALSE	TRUE
ctable_show_label	<logical>	Display Curve Labels	TRUE, FALSE	TRUE
ctable_show_model	<logical>	Display Files / Models	TRUE, FALSE	TRUE
ctable_show_type	<logical>	Display Entity Types	TRUE, FALSE	TRUE
ctable_show_entity_id	<logical>	Display Entity Ids	TRUE, FALSE	TRUE
ctable_show_component	<logical>	Display Components	TRUE, FALSE	TRUE
ctable_show_style	<logical>	Display Curve Styles	TRUE, FALSE	TRUE
ctable_show_directory	<logical>	Display Directories	TRUE, FALSE	TRUE

The following options control the default location and name of where T/HIS looks for model database files.

Preference	Type	Description	Valid arguments	Default
database_dir	<string>	Directory to look in for model database (XML) files		<none>
database_file	<string>	Default model database (XML) file		<none>
database_expand	<integer>	Number of levels to automatically expand in model database tree (-1 ALL)	-1 - 2147483646	0

The following strings and values control display options

Preference	Type	Description	Valid arguments	Default
axis_width	<real>	Default line width for axis (pixels)	1.0, 2.0, 4.0, 8.0	2.0

axis_colour	<string>	Axis colour	FOREGROUND, BACKGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
axis_top	<string>	Turn ON/OFF drawing of graph top axis	ON, OFF	ON
axis_right	<string>	Turn ON/OFF drawing of graph right axis	ON, OFF	ON
border_on	<logical>	Display border	TRUE, FALSE	TRUE
border_width	<real>	Default line width for border (pixels)	1.0, 2.0, 4.0, 8.0	1.0
border_colour	<string>	Border colour	FOREGROUND, BACKGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
grid_on	<logical>	Display grid	TRUE, FALSE	FALSE
grid_width	<real>	Default line width for grid (pixels)	1.0, 2.0, 4.0, 8.0	2.0

The following strings and values control formatting of values for graphs

Preference	Type	Description	Valid arguments	Default
add_exponent_to_label	<logical>	Add axis multiplier to label	TRUE, FALSE	TRUE
x_axis_decimal_places	<string>	Number of decimal places displayed for X axis values	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, Default(3)	Default(3)
x_axis_format	<string>	Format used to display X axis values	Automatic, General, Scientific, Default(Automatic)	Default(Automatic)
y_axis_decimal_places	<string>	Number of decimal places displayed for Y axis values	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, Default(3)	Default(3)

y_axis_format	<string>	Format used to display Y axis values	Automatic, General, Scientific, Default(Automatic)	Default(Automatic)
y2_axis_decimal_places	<string>	Number of decimal places displayed for second Y axis values	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, Default(3)	Default(3)
y2_axis_format	<string>	Format used to display second Y axis values	Automatic, General, Scientific, Default(Automatic)	Default(Automatic)
<b>colours</b>				
background_colour	<string>	Background colour	WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	BLACK
foreground_colour	<string>	Foreground colour	WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	WHITE
user_colour1	<string>	User defined colour 1 (HEX RRGGBB value)		<none>
user_colour2	<string>	User defined colour 2 (HEX RRGGBB value)		<none>
user_colour3	<string>	User defined colour 3 (HEX RRGGBB value)		<none>
user_colour4	<string>	User defined colour 4 (HEX RRGGBB value)		<none>

user_colour5	<string>	User defined colour 5 (HEX RRGGBB value)	<none>
user_colour6	<string>	User defined colour 6 (HEX RRGGBB value)	<none>

The following options control the preferred order of [data sources](#) for various entities

Preference	Type	Description	Valid arguments	Default
use_elout	<logical>	Use ELOUT in preference to ELOUTDET for Shell and ThickShell data components from LSDA file	TRUE, FALSE	FALSE
global	<ordered>	Data source for global data	LSDA, ASCII, THF, none	<none>
part	<ordered>	Data source for part data	THF, LSDA, ASCII, none	<none>
node	<ordered>	Data source for node data	THF, LSDA, ASCII, none	<none>
<b>elements</b>				
solid	<ordered>	Data source for solid data	THF, LSDA, none	<none>
beam	<ordered>	Data source for beam data	THF, LSDA, none	<none>
shell	<ordered>	Data source for shell data	THF, LSDA, none	<none>
tshell	<ordered>	Data source for thick shell data	THF, LSDA, none	<none>
spring	<ordered>	Data source for spring data	XTF, LSDA, ASCII, none	<none>
seatbelt	<ordered>	Data source for seatbelt data	XTF, LSDA, ASCII, none	<none>
retractor	<ordered>	Data source for retractor data	XTF, LSDA, ASCII, none	<none>
slipping	<ordered>	Data source for slipping data	XTF, LSDA, ASCII, none	<none>
wall	<ordered>	Data source for rigid wall data	XTF, LSDA, ASCII, none	<none>
contact	<ordered>	Data source for contact data	XTF, LSDA, ASCII, none	<none>
reaction	<ordered>	Data source for nodal reaction data	XTF, LSDA, ASCII, none	<none>
airbag	<ordered>	Data source for airbag data	XTF, LSDA, ASCII, none	<none>
joint	<ordered>	Data source for joint data	LSDA, ASCII, none	<none>
section	<ordered>	Data source for section data	LSDA, ASCII, none	<none>
subsystem	<ordered>	Data source for subsystems data	LSDA, ASCII, none	<none>
geo_contact	<ordered>	Data source for geometric contact data	LSDA, ASCII, none	<none>
nodal_rb	<ordered>	Data source for nodal rigid body data	LSDA, ASCII, none	<none>
weld	<ordered>	Data source for spotweld data	LSDA, ASCII, none	<none>
spc	<ordered>	Data source for spc data	LSDA, ASCII, none	<none>
boundary	<ordered>	Data source for boundary data	LSDA, ASCII, none	<none>
fsi	<ordered>	Data source for fluid structural interaction data	LSDA, ASCII, none	<none>
sph	<ordered>	Data source for SPH data	LSDA, ASCII, none	<none>
tracer	<ordered>	Data source for TRACER data	LSDA, ASCII, none	<none>
pulley	<ordered>	Data source for PULLEY data	LSDA, ASCII, none	<none>

The following strings and values control [axes, title, and legend formatting](#) for graphs

Preference	Type	Description	Valid arguments	Default
title_size	<string>	Font size for title	8, 10, 12, 14, 18, 24, Default	Default
title_font	<string>	Font for title	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default

title_colour	<string>	Colour of title	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
x_label_size	<string>	Font size for X axis label	8, 10, 12, 14, 18, 24, Default	Default
x_label_font	<string>	Font for X axis label	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default
x_label_colour	<string>	Colour of X axis label	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
x_axis_size	<string>	Font size for X axis units	8, 10, 12, 14, 18, 24, Default	Default
x_axis_font	<string>	Font for X axis units	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default
x_axis_colour	<string>	Colour of X axis units	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
y_label_size	<string>	Font size for Y axis label	8, 10, 12, 14, 18, 24, Default	Default
y_label_font	<string>	Font for Y axis label	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default
y_label_colour	<string>	Colour of Y axis label	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
y_axis_size	<string>	Font size for Y axis units	8, 10, 12, 14, 18, 24, Default	Default

y_axis_font	<string>	Font for Y axis units	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default
y_axis_colour	<string>	Colour of Y axis units	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
y2_label_size	<string>	Font size for second Y axis label	8, 10, 12, 14, 18, 24, Default	Default
y2_label_font	<string>	Font for second Y axis label	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default
y2_label_colour	<string>	Colour of second Y axis label	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
y2_axis_size	<string>	Font size for second Y axis units	8, 10, 12, 14, 18, 24, Default	Default
y2_axis_font	<string>	Font for second Y axis units	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default
y2_axis_colour	<string>	Colour of second Y axis units	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
legend_size	<string>	Font size for curve legends	8, 10, 12, 14, 18, 24, Default	Default
legend_font	<string>	Font for second curve legends	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default



legend_colour	<string>	Colour of curve legends	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
---------------	----------	-------------------------	---	------------

The following strings and values control how T/HIS starts

Preference	Type	Description	Valid arguments	Default
graphics_type	<string>	Graphics format to start T/HIS with	OPENGL, TTY, DEFAULT	OPENGL
maximise	<logical>	Maximise window when T/HIS started	TRUE, FALSE	FALSE
placement	<string>	Location for initial window on multi-screen display	LEFT, RIGHT, BOTTOM, TOP, LEFT_BOTTOM, LEFT_TOP, RIGHT_BOTTOM, RIGHT_TOP	<none>
image_format	<string>	Default image format	BMP_8_C, BMP_8_UN, PNG_8, GIF_8, BMP_24_UN, PNG_24, JPG_24, PPM_24	JPG_24
white_background_image	<logical>	Write images with white background	TRUE, FALSE	FALSE
rhs_number_columns	<integer>	Number of columns of Tools buttons	4 - 50	4

The following strings and values control laser plotting setup

Preference	Type	Description	Valid arguments	Default
laser_paper_size	<string>	Default paper size	A4, A3, US	A4
laser_orientation	<string>	Default page orientation	Portrait, Landscape	Landscape
laser_top_margin	<real>	Top margin size in mm		10
laser_bottom_margin	<real>	Bottom margin size in mm		30
laser_left_margin	<real>	Left margin size in mm		20
laser_right_margin	<real>	Right margin size in mm		10

The following options affect the appearance and behaviour of the graphical user interface, left handed support, and the mouse

Preference	Type	Description	Valid arguments	Default
display_factor	<real>	Factor on display size (0.5-2.0)	0.5 - 2.0	1.0
display_brightness	<real>	Menu brightness (0.0-1.0)	0.0 - 1.0	1.0
display_saturation	<real>	Menu colour saturation (0.0-1.0)	0.0 - 1.0	1.0
button_gradation	<real>	Button shade gradation (0.0-1.0)	0.0 - 1.0	0.5
dv_sync_windows	<string>	Dyn view method(s) for synchronising windows	ICON, ICON+CAPS, ICON+NUM, ICON+CAPS+NUM	ICON+CAPS
dv_left_shift	<string>	Dyn view action for shift + Left mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ROTATION_XYZ
dv_middle_shift	<string>	Dyn view action for shift + Middle mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	TRANSLATION

dv_right_shift	<string>	Dyn view action for shift + Right mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ZOOM_UP_+VE
dv_left_ctrl	<string>	Dyn view action for ctrl + Left mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ROTATION_XYZ
dv_middle_ctrl	<string>	Dyn view action for ctrl + Middle mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	TRANSLATION
dv_right_ctrl	<string>	Dyn view action for ctrl + Right mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ZOOM_UP_+VE
dv_left_both	<string>	Dyn view action for shift+ctrl + Left mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ROTATION_XYZ
dv_middle_both	<string>	Dyn view action for shift+ctrl + Middle mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	TRANSLATION
dv_right_both	<string>	Dyn view action for shift+ctrl + Right mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ZOOM_UP_+VE
dv_shift_action	<string>	Dynamic viewing mode for shift + mouse button	CURRENT, WIREFRAME, FREE_EDGE, UNUSED	CURRENT
dv_ctrl_action	<string>	Dynamic viewing mode for ctrl + mouse button	CURRENT, WIREFRAME, FREE_EDGE, UNUSED	WIREFRAME
dv_both_action	<string>	Dynamic viewing mode for shift+ctrl + mouse button	CURRENT, WIREFRAME, FREE_EDGE, UNUSED	FREE_EDGE
font_scaling	<logical>	whether text in GUI buttons can be scaled down to fit	TRUE, FALSE	TRUE
font_silent	<logical>	whether to write explanatory text if wanted fonts are not found	TRUE, FALSE	FALSE
font_size	<string>	Menu font size	SMALL, DEFAULT, LARGE	DEFAULT
font_type	<string>	Menu font typeface and strength	HELVETICA, HELVETICA-BOLD, TIMES, TIMES-BOLD, COURIER, COURIER-BOLD	HELVETICA
left_handed	<string>	Left handed switching of mouse and/or keyboard	NONE, MOUSE, KEYBOARD, ALL	NONE
zoom_factor	<real>	Zoom Factor for mouse wheel (0.01-0.2)	0.01 - 0.2	0.05
czoom_factor	<real>	Factor for right mouse dynamic zoom (0.01-0.2)	0.01 - 0.2	0.05
kzoom_factor	<real>	Factor for +/- keyboard short-cut keys	0.01 - 100.0	2.0

The following options define how Javascripts are processed by THIS. See [section 5.23](#) for further details.

Preference	Type	Description	Valid arguments	Default
script_directory	<string>	Directory in which T/HIS looks for scripts		\$OA_INSTALL/this_library/scripts
javascript_memory_size	<integer>	Maximum memory allocated for garbage collection		25

Keys can have functions assigned to them:

Preference	Type	Description	Valid arguments	Default
F1_key	<string>	Shortcut for F1		<none>
F2_key	<string>	Shortcut for F2		<none>
F3_key	<string>	Shortcut for F3		<none>
F4_key	<string>	Shortcut for F4		<none>
F5_key	<string>	Shortcut for F5		<none>
F6_key	<string>	Shortcut for F6		<none>
F7_key	<string>	Shortcut for F7		<none>
F8_key	<string>	Shortcut for F8		<none>
F9_key	<string>	Shortcut for F9		<none>
F10_key	<string>	Shortcut for F10		<none>
F11_key	<string>	Shortcut for F11		<none>
F12_key	<string>	Shortcut for F12		<none>
A_key	<string>	Shortcut for A		AUTOSCALE
B_key	<string>	Shortcut for B		BLANK
C_key	<string>	Shortcut for C		CURVE_MENU
D_key	<string>	Shortcut for D		DATUM_MENU
E_key	<string>	Shortcut for E		<none>
F_key	<string>	Shortcut for F		FAST_TCF_MENU
G_key	<string>	Shortcut for G		NEW_WINDOW
H_key	<string>	Shortcut for H		<none>
I_key	<string>	Shortcut for I		<none>
J_key	<string>	Shortcut for J		JAVASCRIPT_MENU
K_key	<string>	Shortcut for K		<none>
L_key	<string>	Shortcut for L		<none>
M_key	<string>	Shortcut for M		<none>
N_key	<string>	Shortcut for N		EDIT_NEXT
O_key	<string>	Shortcut for O		<none>
P_key	<string>	Shortcut for P		PLOT
Q_key	<string>	Shortcut for Q		QUICK_PICK
R_key	<string>	Shortcut for R		REVERSE

S_key	<string>	Shortcut for S		<none>
T_key	<string>	Shortcut for T		TIDY_MENUS
U_key	<string>	Shortcut for U		UNBLANK
V_key	<string>	Shortcut for V		CURVE_GROUP
W_key	<string>	Shortcut for W		<none>
X_key	<string>	Shortcut for X		CURVE_TABLE
Y_key	<string>	Shortcut for Y		Y_AUTOSCALE
Z_key	<string>	Shortcut for Z		ZOOM
a_key	<string>	Shortcut for a		AUTOSCALE
b_key	<string>	Shortcut for b		BLANK
c_key	<string>	Shortcut for c		CURVE_MENU
d_key	<string>	Shortcut for d		DATUM_MENU
e_key	<string>	Shortcut for e		<none>
f_key	<string>	Shortcut for f		FAST_TCF_MENU
g_key	<string>	Shortcut for g		NEW_WINDOW
h_key	<string>	Shortcut for h		<none>
i_key	<string>	Shortcut for i		<none>
j_key	<string>	Shortcut for j		JAVASCRIPT_MENU
k_key	<string>	Shortcut for k		<none>
l_key	<string>	Shortcut for l		<none>
m_key	<string>	Shortcut for m		<none>
n_key	<string>	Shortcut for n		EDIT_NEXT
o_key	<string>	Shortcut for o		<none>
p_key	<string>	Shortcut for p		PLOT
q_key	<string>	Shortcut for q		QUICK_PICK
r_key	<string>	Shortcut for r		REVERSE
s_key	<string>	Shortcut for s		<none>
t_key	<string>	Shortcut for t		TIDY_MENUS
u_key	<string>	Shortcut for u		UNBLANK
v_key	<string>	Shortcut for v		CURVE_GROUP
w_key	<string>	Shortcut for w		<none>
x_key	<string>	Shortcut for x		CURVE_TABLE

y_key	<string>	Shortcut for y		Y_AUTOSCALE
z_key	<string>	Shortcut for z		ZOOM
SPACE_key	<string>	Shortcut for space		PLOT
ZERO_key	<string>	Shortcut for 0		COPY_AXIS
ONE_key	<string>	Shortcut for 1		TILE_TALL
TWO_key	<string>	Shortcut for 2		TILE_WIDE
THREE_key	<string>	Shortcut for 3		CASCADE
FOUR_key	<string>	Shortcut for 4		LAYOUT_1X1
FIVE_key	<string>	Shortcut for 5		LAYOUT_2X2
SIX_key	<string>	Shortcut for 6		LAYOUT_3X3
SEVEN_key	<string>	Shortcut for 7		<none>
EIGHT_key	<string>	Shortcut for 8		<none>
NINE_key	<string>	Shortcut for 9		<none>
EXCLAMATION_key	<string>	Shortcut for !		<none>
DOUBLEQUOTE_key	<string>	Shortcut for "		<none>
HASH_key	<string>	Shortcut for #		<none>
DOLLAR_key	<string>	Shortcut for \$		<none>
PERCENT_key	<string>	Shortcut for %		<none>
AMPERSAND_key	<string>	Shortcut for &		<none>
SINGLEQUOTE_key	<string>	Shortcut for '		<none>
LEFTBRACKET_key	<string>	Shortcut for (		<none>
RIGHTBRACKET_key	<string>	Shortcut for )		<none>
ASTERISK_key	<string>	Shortcut for *		<none>
PLUS_key	<string>	Shortcut for +		ZOOM_IN
COMMA_key	<string>	Shortcut for ,		<none>
MINUS_key	<string>	Shortcut for -		ZOOM_OUT
DOT_key	<string>	Shortcut for .		<none>
SLASH_key	<string>	Shortcut for /		SHORTCUT
COLON_key	<string>	Shortcut for :		<none>
SEMICOLON_key	<string>	Shortcut for ;		<none>
LESSTHAN_key	<string>	Shortcut for <		<none>
EQUALS_key	<string>	Shortcut for =		ZOOM_IN

GREATERTHAN_key	<string>	Shortcut for >		<none>
QUESTIONMARK_key	<string>	Shortcut for ?		SHORTCUT
AT_key	<string>	Shortcut for @		<none>
LEFTSQUAREBRACKET_key	<string>	Shortcut for [		<none>
BACKSLASH_key	<string>	Shortcut for \		<none>
RIGHTSQUAREBRACKET_key	<string>	Shortcut for ]		<none>
CIRCUMFLEX_key	<string>	Shortcut for ^		<none>
UNDERSCORE_key	<string>	Shortcut for _		ZOOM_OUT
BACKTICK_key	<string>	Shortcut for ‘		<none>
LEFTCURLYBRACKET_key	<string>	Shortcut for {		<none>
PIPE_key	<string>	Shortcut for		<none>
RIGHTCURLYBRACKET_key	<string>	Shortcut for }		<none>
TILDE_key	<string>	Shortcut for ~		<none>

The following strings control the T/HIS header and version number at the bottom right of the plot space

Preference	Type	Description	Valid arguments	Default
user_text_line_1	<string>	Text for line 1		<none>
user_text_line_2	<string>	Text for line 2		<none>
user_text_line_3	<string>	Text for line 3		<none>
user_text_line_4	<string>	Text for line 4		<none>
user_text_line_5	<string>	Text for line 5		<none>
user_text_line_6	<string>	Text for line 6		<none>
user_text_size_1	<string>	Size of text on line 1	8, 10, 12, 14, 18, 24, Default	Default
user_text_size_2	<string>	Size of text on line 2	8, 10, 12, 14, 18, 24, Default	Default
user_text_size_3	<string>	Size of text on line 3	8, 10, 12, 14, 18, 24, Default	Default
user_text_size_4	<string>	Size of text on line 4	8, 10, 12, 14, 18, 24, Default	Default
user_text_size_5	<string>	Size of text on line 5	8, 10, 12, 14, 18, 24, Default	Default
user_text_size_6	<string>	Size of text on line 6	8, 10, 12, 14, 18, 24, Default	Default
user_text_font	<string>	Font for user text	Helvetica_Medium, Helvetica_Bold, Courier_Medium, Courier_Bold, Times_Medium, Times_bold, Default	Default

user_text_colour	<string>	Colour for user text	FOREGROUND, WHITE, BLACK, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, ORANGE, TURQUOISE, INDIGO, LIME, SKY, PINK, PALE_YELLOW, GOLD, OLIVE, DARK_MAGENTA, MEDIUM_GREEN, MEDIUM_BLUE, HOT_PINK, LIGHT_PINK, SEA_GREEN, MAROON, DARK_GREEN, PURPLE, NAVY, DARK_GREY, MEDIUM_GREY, LIGHT_GREY, USER_1, USER_2, USER_3, USER_4, USER_5, USER_6, COL_1, COL_2, COL_3, COL_4, COL_5, COL_6, COL_7, COL_8, COL_9, COL_10, COL_11, COL_12, COL_13, COL14, COL_15, COL_16, COL_17, COL_18, COL_19, COL_20, COL_21, COL_22, COL_23, COL_24, COL_25, COL_26, COL_27, COL_28, COL_29, COL_30, COL_31, COL_32, COL_33, COL_34, COL_35, COL_36	FOREGROUND
------------------	----------	----------------------	---	------------

The following control treatment of unicode

Preference	Type	Description	Valid arguments	Default
cjk_unix_font	<string>	Font to use for CJK text on unix machines		-misc-fixed-medium-r-normal-* <sub>12</sub> -* <sub>12</sub> -* <sub>12</sub> -* <sub>12</sub> -*
cjk_windows_font	<string>	Font to use for CJK text on windows machines		MS Gothic 12
file_encoding	<string>	Character encoding for script files	Latin-1, BIG5, EUC-CN, EUC-JP, EUC-KR, GB, GBK, ISO-2022-CN, ISO-2022-CN-EXT, ISO-2022-JP, ISO-2022-JP-2, ISO-2022-KR, JOHAB, Shift-JIS, UTF-8, UTF-16BE, UTF-16LE, UTF-16, UTF-32BE, UTF-32LE, UTF-32	Latin-1

The following strings and values control the display of UNIT information in T/HIS

Preference	Type	Description	Valid arguments	Default
model_units	<string>	Sets the default UNIT system for models	U1 m:kg:s (SI), U2 mm:T:s, U3 mm:kg:ms, U4 mm:gm:ms, U5 ft:slug:s, U6 m:T:s	U1 m:kg:s (SI)
display_units	<string>	Sets the default UNIT system used to display results	U1 m:kg:s (SI), U2 mm:T:s, U3 mm:kg:ms, U4 mm:gm:ms, U5 ft:slug:s, U6 m:T:s	U1 m:kg:s (SI)
write_csv_units	<logical>	Write UNIT information to CSV files	TRUE, FALSE	TRUE

The following is an example file. Note that blank lines and lines not beginning **<programme name>:** are ignored, so comment lines may be added if required. However, lines to be read must start at column 1 and there must not be any gaps in the keyword.

```

this*laser_paper_size:      A4
this*laser_orientation:    Landscape
this*laser_mode:           Greyscale
this*user_text_line_2:     Design Project
this*user_text_line_4:     Phase II Results

```

The user preferences option is not available in command line mode, however the **oa\_pref** file is read and applied. Setting paper size and margins can therefore only be done through this method.

## Global preferences.

From version 9.3 onwards global preferences that apply to all programs can be specified using **"oasys"** as the program name.

**oasys\*<keyword>: <argument>**

At present the following global preferences can be defined

If a preference is defined twice using both **"oasys"** and **"this"** then the **"this"** setting will override the global setting.

Preference	Type	Description	Valid arguments	Default
file_names	<string>	Controls input filename syntax. LSTC = d3*, OASYS = job.ptf*	OASYS, LSTC	OASYS
html_application	<string>	Location of HTML browser		<none>
image_format	<string>	Default image format	BMP_8_C, BMP_8_UN, PNG_8, GIF_8, BMP_24_UN, PNG_24, JPG_24, PPM_24	JPG_24
maximise	<logical>	Maximise window when Program is started	TRUE, FALSE	FALSE
placement	<string>	Location for initial window on multi-screen display	LEFT, RIGHT, BOTTOM, TOP, LEFT_BOTTOM, LEFT_TOP, RIGHT_BOTTOM, RIGHT_TOP	<none>
pdf_application	<string>	Location of PDF browser		<none>
start_in	<string>	Directory to start Program in		<none>

The following control directories

Preference	Type	Description	Valid arguments	Default
home_dir	<string>	"home" directory for user		<none>
install_dir	<string>	Directory Oasys Ltd software is installed in		<none>
manuals_dir	<string>	Directory user manuals are installed in		<none>
temp_dir	<string>	temporary directory for user		<none>
checkpoint_dir	<string>	Directory for checkpoint files, or "none" to suppress them altogether		<none>

The following control laser options

Preference	Type	Description	Valid arguments	Default
laser_paper_size	<string>	Default paper size	US, A4	A4
laser_orientation	<string>	Default page orientation	Portrait, Landscape	Landscape
laser_top_margin	<real>	Top margin size in mm		10
laser_bottom_margin	<real>	Bottom margin size in mm		30
laser_left_margin	<real>	Left margin size in mm		20
laser_right_margin	<real>	Right margin size in mm		10

The following control menu and mouse attributes

Preference	Type	Description	Valid arguments	Default
display_factor	<real>	Factor on display size (0.5-2.0)	0.5 - 2.0	1.0
display_brightness	<real>	Menu brightness (0.0-1.0)	0.0 - 1.0	1.0
display_saturation	<real>	Menu colour saturation (0.0-1.0)	0.0 - 1.0	1.0
button_gradation	<real>	Button shade gradation (0.0-1.0)	0.0 - 1.0	0.5
dv_sync_windows	<string>	Dyn view method(s) for synchronising windows	ICON, ICON+CAPS, ICON+NUM, ICON+CAPS+NUM	ICON+CAPS



dv_left_shift	<string>	Dyn view action for shift + Left mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ROTATION_XYZ
dv_middle_shift	<string>	Dyn view action for shift + Middle mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	TRANSLATION
dv_right_shift	<string>	Dyn view action for shift + Right mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ZOOM_UP_+VE
dv_left_ctrl	<string>	Dyn view action for ctrl + Left mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ROTATION_XYZ
dv_middle_ctrl	<string>	Dyn view action for ctrl + Middle mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	TRANSLATION
dv_right_ctrl	<string>	Dyn view action for ctrl + Right mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ZOOM_UP_+VE
dv_left_both	<string>	Dyn view action for shift+ctrl + Left mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ROTATION_XYZ
dv_middle_both	<string>	Dyn view action for shift+ctrl + Middle mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	TRANSLATION
dv_right_both	<string>	Dyn view action for shift+ctrl + Right mouse	ROTATION_XYZ, ROTATION_XY, ROTATION_Z, ROTATION_SPHERE, TRANSLATION, ZOOM_UP_+VE, ZOOM_DOWN_+VE, UNUSED	ZOOM_UP_+VE
dv_shift_action	<string>	Dynamic viewing mode for shift + mouse button	CURRENT, WIREFRAME, FREE_EDGE, UNUSED	CURRENT
dv_ctrl_action	<string>	Dynamic viewing mode for ctrl + mouse button	CURRENT, WIREFRAME, FREE_EDGE, UNUSED	WIREFRAME
dv_both_action	<string>	Dynamic viewing mode for shift+ctrl + mouse button	CURRENT, WIREFRAME, FREE_EDGE, UNUSED	FREE_EDGE
font_scaling	<logical>	whether text in GUI buttons can be scaled down to fit	TRUE, FALSE	TRUE
font_size	<string>	Menu font size	SMALL, DEFAULT, LARGE	DEFAULT
font_type	<string>	Menu font typeface and strength	HELVETICA, HELVETICA-BOLD, TIMES, TIMES-BOLD, COURIER, COURIER-BOLD	HELVETICA
left_handed	<string>	Left handed switching of mouse and/or keyboard	NONE, MOUSE, KEYBOARD, ALL	NONE
zoom_factor	<real>	Zoom Factor for mouse wheel (0.01-0.2)	0.01 - 0.2	0.05
czoom_factor	<real>	Factor for right mouse dynamic zoom (0.01-0.2)	0.01 - 0.2	0.05
kzoom_factor	<real>	Factor for +/- keyboard short-cut keys	0.01 - 100.0	2.0
mouse_3d_rotation_factor	<real>	Factor applied to the speed of rotation when using a 3D mouse		1.0

mouse_3d_pan_factor	<real>	Factor applied to the speed of panning when using a 3D mouse		1.0
mouse_3d_zoom_factor	<real>	Factor applied to the speed of zooming when using a 3D mouse		1.0

The following control treatment of unicode

Preference	Type	Description	Valid arguments	Default
cjk_unix_font	<string>	Font to use for CJK text on unix machines		-misc-fixed-medium-r-normal-* <sub>12</sub> -*-*-*-* <sub>12</sub>
cjk_windows_font	<string>	Font to use for CJK text on windows machines		MS Gothic 12
file_encoding	<string>	Character encoding for script files	Latin-1, BIG5, EUC-CN, EUC-JP, EUC-KR, GB, GBK, ISO-2022-CN, ISO-2022-CN-EXT, ISO-2022-JP, ISO-2022-JP-2, ISO-2022-KR, JOHAB, Shift-JIS, UTF-8, UTF-16BE, UTF-16LE, UTF-16, UTF-32BE, UTF-32LE, UTF-32	Latin-1

# APPENDIX I - Windows File Associations

## I.1 WINDOWS (PC's)

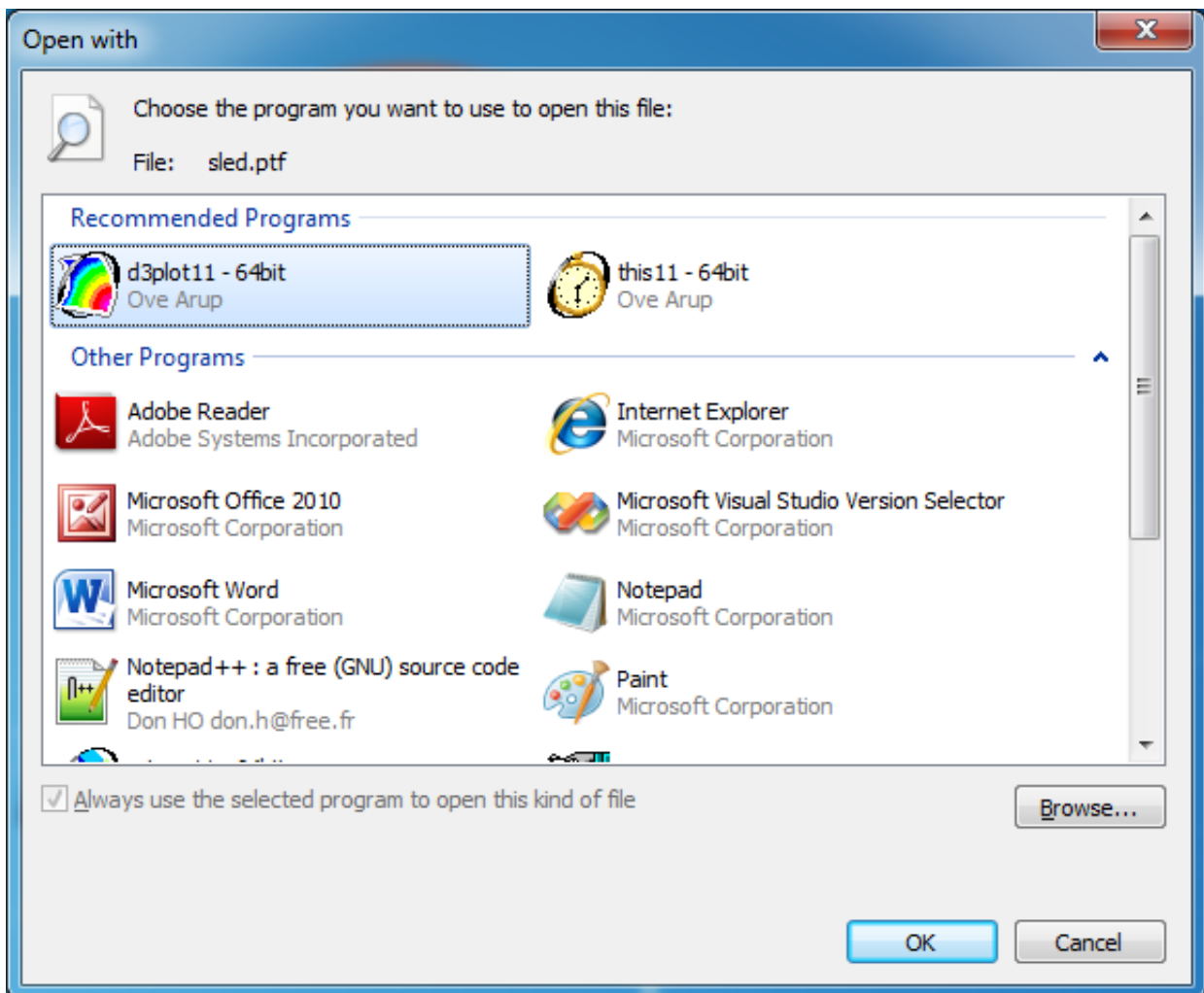
Under Windows on PC it is possible to set up file associations so that double clicking on files with the **.thf**, **.xtf**, **.cur** and **.bdf** extension opens them automatically in T/HIS.

All of these settings are optional: you should be aware that under the Windows operating system associating a filetype (via its extension) with an application is convenient, but can also be restricting and hard to undo.

### I.1.1 To make .thf files open in T/HIS by double-clicking on them

If no application is currently associated with .thf files, a "double-click" won't work, and some non-specific, usually "windows", icon will be displayed with the file.

Right click on any **.thf** file, and select **properties** and then press the Change... tab next to Opens with: from the popup menu.



1. This will bring up the "Open with" panel.
2. Ensure the **Always use...** box is ticked
3. Use the directory browsing window to find the correct T/HIS executable. You are looking for file **this11.exe** or **this11\_x64.exe**.
4. Select the executable and click on **OK** to close the "Open With" window.

T/HIS should now open and read in the selected file and you should now find that:

- All **.thf** files on your system show the T/HIS icon.
- Double-clicking on any such file starts T/HIS and opens that file.

It is not possible to set up the filename "d3thdt" for double-clicking in this way since Windows requires filename extensions when assigning applications to files.)

### I.1.2 To make **.xtf**, **.cur** and **.bdf** files open in T/HIS by double-clicking on them

The procedure is exactly the same as for **.thf** files, and must be carried out for each of the file types that you wish to process by double-clicking:

<b>.xtf</b>	LS-DYNA Extra Time History file
<b>.cur</b>	T/HIS Curve file
<b>.bdf</b>	T/HIS Bulk Data file

**Note that:** File types **.thf** and **.xtf** are opened in this way, but no contents are read in.

File types **.cur** and **.bdf** are opened and their complete contents read in.

## APPENDIX J - T-HIS JavaScript API global class

Description of T/HIS API functions and methods.

The following pages describe the functions ("methods") available in the T/HIS Javascript interface. For information about how to run Javascript files see [Section 5.23](#).

### global class

The global class is the main JavaScript class. [More...](#)

### Class functions

- [AllocateFlag\(\)](#)
- [ClearFlag\(flag\[ \*Flag\* \]\)](#)
- [DialogueInput\(string\\_1, \(string\\_2 ... string\\_n\)\[\*One or more Javascript strings\*\]\)](#)
- [DialogueInputNoEcho\(string\\_1, \(string\\_2 ... string\\_n\)\[\*One or more Javascript strings\*\]\)](#)
- [DisableUpdate\(\)](#)
- [EnableUpdate\(\)](#)
- [ErrorMessage\(string\[\*Any valid javascript type\*\]\)](#)
- [Exit\(\)](#)
- [GetFtcfVar\(name\[\*string\*\]\)](#)
- [Getenv\(name\[\*string\*\]\)](#)
- [Message\(string\[\*Any valid javascript type\*\]\)](#)
- [MilliSleep\(time\[\*integer\*\]\)](#)
- [NumberToString\(number\[\*integer/float\*\], width\[\*integer\*\]\)](#)
- [Plot\(\)](#)
- [Print\(string\[\*Any valid javascript type\*\]\)](#)
- [Println\(string\[\*Any valid javascript type\*\]\)](#)
- [ReturnFlag\(flag\[ \*Flag\* \]\)](#)
- [SetFtcfVar\(name\[\*string\*\]\)](#)
- [Sleep\(time\[\*integer\*\]\)](#)
- [System\(string\[\*Any valid javascript type\*\]\)](#)
- [Unix\(\)](#)
- [WarningMessage\(string\[\*Any valid javascript type\*\]\)](#)
- [Windows\(\)](#)

### Detailed Description

The global class declares the global object in JavaScript that contains the global properties and methods. As well as the core JavaScript methods, T/HIS also defines other additional ones. e.g. [Message\(\)](#), [Print\(\)](#) etc. See the documentation below for more details.

### Details of functions

#### AllocateFlag() [static]

#### Description

Allocate a flag for use in the script. See also [ReturnFlag\(\)](#) and Once allocated the flag is automatically cleared for all the curves currently in T/HIS.

#### Arguments

No arguments

#### Return type

Flag (integer)

## Example

To allocate a flag  

```
var flag = AllocateFlag();
```

---

## ClearFlag(flag/*Flag*) [static]

### Description

Clears a flag on all curves.

### Arguments

Name	Type	Description
flag	<i>Flag</i>	The flag to return.

### Return type

No return value.

### Example

To clear flag f:  

```
ClearFlag(f);
```

---

## DialogueInput(string\_1, (string\_2 ... string\_n)/*One or more Javascript strings*) [static]

### Description

Execute one or more lines of command line dialogue input.

### Arguments

Name	Type	Description
string_1, (string_2 ... string_n)	One or more Javascript strings	The command(s) that are to be executed as if they had been typed into the dialogue box

### Return type

No return value

### Example

To mulitple curves 1 and 2 by 10:  

```
DialogueInputNoEcho("/op mul #1 10 #", "/op mul #2 10 #");
```

Note that each call to DialogueInput starts afresh at the top of the T/HIS command line "tree", so where multiple commands need to be given at sub-menu levels they need to be included in a single call.

---

---

**DialogueInputNoEcho(string\_1, (string\_2 ... string\_n)[*One or more Javascript strings*])**  
**[static]**

## Description

Execute one or more lines of command line dialogue input **with no echo of commands to dialogue box**.

## Arguments

Name	Type	Description
string_1, (string_2 ... string_n)	One or more Javascript strings	The command(s) that are to be executed as if they had been typed into the dialogue box

## Return type

No return value

## Example

To mulitple curves 1 and 2 by 10:

```
DialogueInputNoEcho("/op mul #1 10 #", "/op mul #2 10 #");
```

As with DialogueInput above each call starts at the top of the T/HIS command tree structure, so any commands destined for sub-menus must all be arguments to a single call.

---

## DisableUpdate() [static]

## Description

Temporarily disable the plotting of curves within graphs.

## Arguments

No arguments

## Return type

No return value

## Example

Temporarily disable the plotting of curves within graphs

```
DisableUpdate();
```

---

## EnableUpdate() [static]

## Description

Re-enable the plotting of curves within graphs.

## Arguments

No arguments

## Return type

No return value

---

## Example

Re-enable the plotting of curves within graphs  
`EnableUpdate();`

---

## ErrorMessage(string[*Any valid javascript type*]) [static]

### Description

Print an error message to the dialogue box **adding a carriage return**.

### Arguments

Name	Type	Description
string	Any valid javascript type	The string/item that you want to print

### Return type

No return value

### Example

To print the title of model object m as an error to the dialogue box  
`ErrorMessage("The title is " + m.title);`

---

## Exit() [static]

### Description

Exit script

### Arguments

No arguments

### Return type

No return value

### Example

Exit with  
`Exit();`

---

## GetFtcfVar(name[*string*]) [static]

### Description

Get the value of a FAST-TCF variable

---



## Arguments

Name	Type	Description
name	string	The FAST-TCF variable name (case independent)

## Return type

String containing variable value or null if variable does not exist

## Example

To get the value for FAST-TCF variable Job  

```
var job_name = GetFtcfVar("Job");
```

---

## Getenv(name[*string*]) [static]

## Description

Get the value of an environment variable

## Arguments

Name	Type	Description
name	string	The environment variable name

## Return type

String containing variable value or null if variable does not exist

## Example

To get the value for environment variable HOME  

```
var home = Getenv("HOME");
```

---

## Message(string[*Any valid javascript type*]) [static]

## Description

Print a message to the dialogue box **adding a carriage return**.

## Arguments

Name	Type	Description
string	Any valid javascript type	The string/item that you want to print

## Return type

No return value

## Example

To print the title of model object m as a message to the dialogue box  

```
Message("The title is " + m.title);
```

## MilliSleep(time[integer]) [static]

### Description

Pause execution of the script for *time* milliseconds. See also [Sleep\(\)](#)

### Arguments

Name	Type	Description
time	integer	Number of milliseconds to pause for

### Return type

No return value

### Example

To pause for 500 milliseconds  
`MilliSleep(500);`

---

## NumberToString(number[integer/float], width[integer]) [static]

### Description

Formats a number to a string with the specified width.

### Arguments

Name	Type	Description
number	integer/float	The number you want to format.
width	integer	The width of the string you want to format it to (must be less than 80).

### Return type

String containing the number

### Example

To write the number 1.2345e+6 to a string 10 characters wide  
`var str = NumberToString(1.2345e+6, 10);`

---

## Plot() [static]

### Description

Updates all the T/HIS graphs.

### Arguments

No arguments

### Return type

No return value

---

---

## Example

Update all graphs  
`Plot();`

---

`Print(string[Any valid javascript type])` [static]

## Description

Print a string to stdout. **Note that a carriage return is not added.**

## Arguments

Name	Type	Description
string	Any valid javascript type	The string/item that you want to print

## Return type

No return value

## Example

To print string "Hello, world!"  
`Print("Hello, world!");`

To print the title of model object m with a carriage return  
`print("The title is " + m.title + "\n");`

---

`Println(string[Any valid javascript type])` [static]

## Description

Print a string to stdout **adding a carriage return.**

## Arguments

Name	Type	Description
string	Any valid javascript type	The string/item that you want to print

## Return type

No return value

## Example

To print string "Hello, world!" automatically adding a carriage return  
`Println("Hello, world!");`

To print the title of model object m, automatically adding a carriage return  
`Println("The title is " + m.title);`

---

## ReturnFlag(flag[[Flag](#)]) [static]

### Description

Return a flag used in the script. See also [AllocateFlag\(\)](#) and

### Arguments

Name	Type	Description
flag	<a href="#">Flag</a>	The flag to return.

### Return type

No return value.

### Example

To return flag f:  
`ReturnFlag(f) ;`

---

## SetFtcfVar(name[*string*]) [static]

### Description

Set the value of a FAST-TCF variable. If the variable already exists then it's value is updated

### Arguments

Name	Type	Description
name	string	The FAST-TCF variable name (case independent)

### Return type

String containing variable value or null if variable does not exist

### Example

To create a new FAST-TCF variable called run\_number with the value "10"  
`var home = SetFtcfVar("run_number", "10") ;`

---

## Sleep(time[*integer*]) [static]

### Description

Pause execution of the script for *time* seconds. See also [MilliSleep\(\)](#)

### Arguments

Name	Type	Description
time	integer	Number of seconds to pause for

### Return type

No return value

---

## Example

To pause for 2 seconds  
`Sleep(2);`

---

## System(string[*Any valid javascript type*]) [static]

### Description

Do a system command outside T/HIS.

### Arguments

Name	Type	Description
string	Any valid javascript type	The system command that you want to do

### Return type

integer (probably zero if command successful but is implementation-dependant)

### Example

To make the directory "example"  
`System("mkdir example");`

---

## Unix() [static]

### Description

Test whether script is running on a Unix/Linux operating system. See also [Windows\(\)](#)

### Arguments

No arguments

### Return type

true if Unix/Linux, false if not

### Example

To test if the OS is Unix  
`if ( Unix() )`

---

## WarningMessage(string[*Any valid javascript type*]) [static]

### Description

Print a warning message to the dialogue box **adding a carriage return**.

---

## Arguments

Name	Type	Description
string	Any valid javascript type	The string/item that you want to print

## Return type

No return value

## Example

To print the title of model object m as a warning to the dialogue box  
`WarningMessage("The title is " + m.title);`

---

## Windows() [static]

## Description

Test whether script is running on a Windows operating system. See also [Unix\(\)](#)

## Arguments

No arguments

## Return type

true if Windows, false if not

## Example

To test if the OS is Windows  
`if ( Windows() )`

---

## Colour class

The Colour class contains constants relating to colours. [More...](#)

### Colour constants

Name	Description
Colour.BACKGROUND	Background colour
Colour.BLACK	Colour black
Colour.BLUE	Colour blue
Colour.CYAN	Colour cyan
Colour.DARK_GREEN	Colour dark green
Colour.DARK_GREY	Colour dark grey
Colour.DARK_MAGENTA	Colour dark magenta
Colour.FOREGROUND	Foreground colour
Colour.GOLD	Colour gold
Colour.GREEN	Colour green
Colour.HOT_PINK	Colour hot pink
Colour.INDIGO	Colour indigo
Colour.LIGHT_GREY	Colour light grey
Colour.LIGHT_PINK	Colour light pink
Colour.LIME	Colour lime
Colour.MAGENTA	Colour magenta
Colour.MAROON	Colour maroon
Colour.MEDIUM_BLUE	Colour medium blue
Colour.MEDIUM_GREEN	Colour medium green
Colour.MEDIUM_GREY	Colour medium grey
Colour.NAVY	Colour navy
Colour.OLIVE	Colour olive
Colour.ORANGE	Colour orange
Colour.PALE_YELLOW	Colourpale yellow
Colour.PINK	Colour pink
Colour.PURPLE	Colour purple
Colour.RED	Colour red
Colour.SEA_GREEN	Colour sea green
Colour.SKY	Colour sky
Colour.TURQUOISE	Colour turquoise
Colour.USER_1	Colour user defined 1
Colour.USER_2	Colour user defined 2
Colour.USER_3	Colour user defined 3
Colour.USER_4	Colour user defined 4

Colour.USER_5	Colour user defined 5
Colour.USER_6	Colour user defined 6
Colour.WHITE	Colour white
Colour.YELLOW	Colour yellow

## Detailed Description

The Colour class is used to define colours:  
`p.colour = Colour.RED;`



## Curve class

The Curve class gives you access to curves in T/HIS. [More...](#)

### Class functions

- [Copy](#)(source[integer], target[integer])
- [Delete](#)(curve[integer])
- [Exists](#)(curve[integer])
- [First](#)()
- [FirstFreeID](#)()
- [FirstID](#)()
- [FlagAll](#)(flag[integer])
- [GetFromID](#)(ID[integer])
- [GetFromTag](#)(TAG[string])
- [HighestID](#)()
- [Pick](#)(prompt[string], modal (optional)[boolean])
- [Select](#)(flag[integer], prompt[string], modal (optional)[boolean])
- [UnflagAll](#)(flag[integer])

### Member functions

- [AddPoint](#)(xvalue[real], yvalue[real])
- [AddToGraph](#)(graph, graph...[int])
- [ClearFlag](#)(flag[integer])
- [DeletePoint](#)(ipt[integer])
- [Flagged](#)(flag[integer])
- [GetPoint](#)(row[integer])
- [InsertPoint](#)(ipt[integer], xvalue[real], yvalue[real], position[integer])
- [Next](#)()
- [Previous](#)()
- [RemoveFromGraph](#)(graph, graph...[int])
- [SetFlag](#)(flag[integer])
- [SetPoint](#)(ipt[integer], xvalue[real], yvalue[real])
- [Update](#)()

### Curve constants

Name	Description
Curve.AFTER	Insertion of curve data option.
Curve.BEFORE	Insertion of curve data option.

### Curve properties

Name	Type	Description
average	float	Curve average value (read only)
colour	<a href="#">Colour</a>	The colour of the curve
directory	string	Directory the curve came from
entity_id	integer	The ID of the entity that the curve was generated from.
entity_type	<a href="#">Entity Type</a>	The entity type that the curve was generated from
file	string	Filename the curve came from
hic	float	Curve HIC value - returns 0.0 if the HIC hasn't been calculated (read only)
hic_tmax	float	End of HIC time windows - returns 0.0 if the HIC hasn't been calculated (read only)
hic_tmin	float	Start of HIC time windows - returns 0.0 if the HIC hasn't been calculated (read only)
hisd	float	Curve HIC(d) value - returns 0.0 if the HIC(d) hasn't been calculated (read only)

hicc_tmax	float	End of HIC(d) time windows - returns 0.0 if the HIC(d) hasn't been calculated (read only)
hicc_tmin	float	Start of HIC(d) time windows - returns 0.0 if the HIC(d) hasn't been calculated (read only)
id	integer	Curve ID (read only)
label	string	Curve label
model	integer	The ID of the model that a curve was read from.
npoints	integer	Number of curve points (read only)
rms	float	Curve RMS value (read only)
style	<a href="#">LineStyle</a>	The line style used to draw the curve
symbol	<a href="#">Symbol</a>	The symbol style for a curve
tag	string	Curve tag. If a FAST-TCF script is running then this is the FAST-TCF tag
title	string	Curve title
tms	float	3ms Clip value - returns 0.0 if the 3ms Clip value hasn't been calculated (read only)
tms_tmax	float	End of 3ms clip time windows - returns 0.0 if the 3ms Clip hasn't been calculated (read only)
tms_tmin	float	Start of 3ms clip time windows - returns 0.0 if the 3ms Clip hasn't been calculated (read only)
unit_system	<a href="#">UnitSystem</a>	The Curve unit system
width	<a href="#">LineWidth</a>	The line width used to draw the curve
x_at_ymax	float	X axis value at the Y axis maximum (read only)
x_at_ymin	float	X axis value at the Y axis minimum (read only)
x_axis_label	string	Curve X axis label
x_axis_unit	<a href="#">Units</a>	The X axis unit
xmax	float	X axis maximum value (read only)
xmin	float	X axis minimum value (read only)
y_axis_label	string	Curve Y axis label
y_axis_unit	<a href="#">Units</a>	The Y axis unit
ymax	float	Y axis maximum value (read only)
ymin	float	Y axis minimum value (read only)

## Detailed Description

The Curve class allows you to create, modify, edit and manipulate curves. See the documentation below for more details.

## Constructor

`new Curve(lcid[integer], tag (optional)[string], Line label (optional)[string], X-axis label (optional)[string], Y-axis label (optional)[string])`

## Description

Create a new [Curve](#) object. The curve will be added to all the currently active graphs.

## Arguments

Name	Type	Description
lcid	integer	<a href="#">Curve</a> number
tag (optional)	string	Tag used to reference the curve in FAST-TCF scripts
Line label (optional)	string	Line label for the curve
X-axis label (optional)	string	X-axis label for the curve
Y-axis label (optional)	string	Y-axis label for the curve

## Return type

[Curve](#) object

## Example

To create a new curve with label 200  

```
var l = new Curve(200);
```

## Details of functions

### AddPoint(xvalue[real], yvalue[real])

## Description

Adds a point at the end of the curve.

## Arguments

Name	Type	Description
xvalue	real	The x value of the point.
yvalue	real	The y value of the point.

## Return type

No return value.

## Example

To add the point x=3.5, y=5.5 to curve l:  

```
l.AddPoint(3.5, 5.5);
```

---

### AddToGraph(graph, graph...[int])

## Description

Adds a curve to a graph.

## Arguments

Name	Type	Description
graph, graph...	int	Optional list of graphs to add the curve to, If undefined then the curve is added to all graphs.

---

## Return type

No return value.

### Example

To add a curve (c) to graphs 1 and 3:

```
c.AddToGraph(1,3);
```

To add a curve (c) to all graphs:

```
c.AddToGraph();
```

---

## ClearFlag(flag[integer])

### Description

Clears a flag on the curve.

### Arguments

Name	Type	Description
flag	integer	Flag to clear on the curve

### Return type

No return value

### Example

To clear flag f for curve l:

```
l.ClearFlag(f);
```

---

## Copy(source[integer], target[integer]) [static]

### Description

Copies a curve.

### Arguments

Name	Type	Description
source	integer	ID of curve to copy from
target	integer	ID of curve to copy to

### Return type

No return value

### Example

To copy curve 1 to curve 4:

```
var curve = Curve.Copy(1,4);
```

To copy curve a to curve b,

```
Curve.Copy(a.id,b.id);
```

---

---

**Delete(curve[integer]) [static]****Description**

Deletes a curve

**Arguments**

Name	Type	Description
curve	integer	ID of curve to delete

**Return type**

No return value

**Example**

To delete curve n

`Curve.Delete(n) ;`

---

**DeletePoint(ipt[integer])****Description**

Deletes a point in a curve. The input for the point number should start at 1 for the 1st point not zero.

**Arguments**

Name	Type	Description
ipt	integer	The point you want to insert the data before or after.

**Return type**

No return value.

**Example**

To delete the 3rd point in curve 1:

`1.DeletePoint(3) ;`

---

**Exists(curve[integer]) [static]****Description**

Checks if a curve exists

**Arguments**

Name	Type	Description
curve	integer	ID of curve to check

**Return type**

TRUE if the curve exists, otherwise FALSE

## Example

To check if a curve n exists

```
var exists = Curve.Exists(n);
```

---

## First() [static]

### Description

Returns the first curve.

### Arguments

No arguments

### Return type

Curve object (or null if there are no more curves in the model).

### Example

To get the 1st curve

```
var curve = Curve.First();
```

---

## FirstFreeID() [static]

### Description

Returns the ID of the first free curve.

### Arguments

No arguments

### Return type

ID of first unsued curve.

### Example

To get the ID of the first free curve:

```
var curve = Curve.FirstFreeID();
```

---

## FirstID() [static]

### Description

Returns the ID of the first curve.

### Arguments

No arguments

### Return type

ID of the first curve defined.

---

---

## Example

To get the 1st curve

```
var curve = Curve.FirstID();
```

---

## FlagAll(flag[integer]) [static]

### Description

Flags all of the curves with a defined flag

### Arguments

Name	Type	Description
flag	integer	Flag to set on the curves

### Return type

No return value

### Example

To flag all of the curves with flag f:

```
var curve = Curve.FlagAll(f);
```

---

## Flagged(flag[integer])

### Description

Checks if the curve is flagged or not.

### Arguments

Name	Type	Description
flag	integer	Flag to check on the curve

### Return type

true if flagged, false if not.

### Example

To check if curve d has flag f set on it:

```
if (d.Flagged(f) ) do_something...
```

---

## GetFromID(ID[integer]) [static]

### Description

Returns the curve object for a curve ID.

---

## Arguments

Name	Type	Description
ID	integer	ID of curve to return object for

## Return type

Curve object (or null if the curve does not exist).

## Example

To get the curve n  
`var curve = Curve.GetFromID(n) ;`

---

## GetFromTag(TAG[string]) [static]

## Description

Finds a curve from it's Tag. This function is only available when running a Javascript from within a FAST-TCF script

## Arguments

Name	Type	Description
TAG	string	TAG of curve to return object for

## Return type

Curve object (or null if there are no free curves).

## Example

To get the curve with a tag "tag"  
`var curve = Curve.GetFromTag(tag) ;`

---

## GetPoint(row[integer])

## Description

Returns x and y data for a point in a curve. The input for the point number should start at 1 for the 1st point not zero. In the array returned array[0] contains the x axis value and array[1] contains the y-axis value.

## Arguments

Name	Type	Description
row	integer	The point you want the data for.

## Return type

An array containing the x value and the y value.

---



## Example

To get the curve data for the 3rd point for curve l:

```
if (l.npoints >= 3)
{
    var point_data = l.GetPoint(3);
}
```

## HighestID() [static]

### Description

Returns the ID of the highest curve currently being used

### Arguments

No arguments

### Return type

ID of highest curve currently being used.

## Example

To get the highest curve ID

```
var id= Curve.HighestID();
```

## InsertPoint(ipt[integer], xvalue[real], yvalue[real], position[integer])

### Description

Inserts a new point before or after the specified point.

### Arguments

Name	Type	Description
ipt	integer	The point you want to insert the data before or after.
xvalue	real	The x value of the point.
yvalue	real	The y value of the point.
position	integer	Specify either before or after the selected point. Use 'Curve.BEFORE' for before, and 'Curve.AFTER' for after.

### Return type

No return value.

## Example

To insert the values after the 3rd row to x=3, y=5 for curve l:

```
l.InsertPoint(3, 3, 5, Curve.AFTER);
```

## Next()

### Description

Returns the next curve in the model.

### Arguments

No arguments

### Return type

Curve object (or null if there are no more curves in the model).

### Example

To get the curve in model m after curve l:  
`var curve = l.Next();`

---

## Pick(prompt[*string*], modal (optional)[*boolean*]) [static]

### Description

Picks a single curve.

### Arguments

Name	Type	Description
prompt	string	Text to display as a prompt to the user
modal (optional)	boolean	If selection is modal (blocks the user from doing anything else in T/HIS until this window is dismissed). If omitted the selection will be modal.

### Return type

Curve object (or null if the user cancels the pick operation).

### Example

To pick a curve, giving the prompt 'Pick curve':  
`Curve.Pick('Pick curves');`

---

## Previous()

### Description

Returns the previous curve in the model.

### Arguments

No arguments

### Return type

Curve object (or null if there are no more curves in the model).

---

## Example

To get the curve in model m before this one:  

```
var curve = curve.Previous();
```

## RemoveFromGraph(graph, graph...*[int]*)

### Description

Removes a curve from a graph.

### Arguments

Name	Type	Description
graph, graph...	int	Optional list of graphs to remove the curve from, If undefined then the curve is removed from all graphs.

### Return type

No return value.

### Example

To remove a curve (c) from graphs 1 and 3:  

```
c.RemoveFromGraph(1,3);
```

To remove a curve (c) from all graphs:  

```
c.RemoveFromGraph();
```

## Select(flag*[integer]*, prompt*[string]*, modal (optional)*[boolean]*) [static]

### Description

Allows the user to select curves.

### Arguments

Name	Type	Description
flag	integer	Flag to use when selecting curves
prompt	string	Text to display as a prompt to the user
modal (optional)	boolean	If selection is modal (blocks the user from doing anything else in T/HIS until this window is dismissed). If omitted the selection will be modal.

### Return type

Number of items selected or null if menu cancelled

### Example

To select curves, flagging those selected which flag f, giving the prompt 'Select curves':  

```
Curve.Select(f, 'Select curves');
```

## SetFlag(flag[integer])

### Description

Sets a flag on the curve.

### Arguments

Name	Type	Description
flag	integer	Flag to set on the curve

### Return type

No return value

### Example

To set flag f for curve l:

```
l.SetFlag(f);
```

---

## SetPoint(ipt[integer], xvalue[real], yvalue[real])

### Description

Sets the x and y values for a specified point in a curve.

### Arguments

Name	Type	Description
ipt	integer	The point to set the data for.
xvalue	real	The x value of the point.
yvalue	real	The y value of the point.

### Return type

No return value.

### Example

To set the values for the 3rd point to x=3, y=5 for curve l:

```
l.SetPoint(3, 3, 5);
```

---

## UnflagAll(flag[integer]) [static]

### Description

Unsets a defined flag on all of the curves.

### Arguments

Name	Type	Description
flag	integer	Flag to unset on the curves

---

## Return type

No return value

### Example

To unset the flag f on all of the curves:

```
var curve = Curve.UnflagAll(f);
```

---

## Update()

### Description

Updates a curve properties (min,max, average values etc).

### Arguments

No arguments

### Return type

No return value.

### Example

To update the properties of curve l:

```
l.Update();
```

---

## Datum class

The Datum class gives you access to datums in T/HIS. [More...](#)

### Class functions

- [Delete](#)(datum[*string*])
- [Exists](#)(datum[*string*])
- [First](#)()
- [GetFromAcronym](#)(datum[*string*])

### Member functions

- [AddToGraph](#)(graph, graph...[*int*])
- [Next](#)()
- [RemoveFromGraph](#)(graph, graph...[*int*])

### Datum constants

Name	Description
Datum.CONSTANT_X	Constant X type datum.
Datum.CONSTANT_Y	Constant Y type datum.
Datum.FILL_ABOVE_BELOW	Fill datum above and below.
Datum.FILL_RIGHT_LEFT	Fill datum right and left.
Datum.LABEL_ABOVE_CENTRE	Label position above centre.
Datum.LABEL_ABOVE_LEFT	Label position above left.
Datum.LABEL_ABOVE_RIGHT	Label position above right.
Datum.LABEL_BELOW_CENTRE	Label position below centre.
Datum.LABEL_BELOW_LEFT	Label position below left.
Datum.LABEL_BELOW_RIGHT	Label position below right.
Datum.LABEL_BOTTOM_LEFT	Label position bottom left.
Datum.LABEL_BOTTOM_RIGHT	Label position bottom right.
Datum.LABEL_MIDDLE_LEFT	Label position middle left.
Datum.LABEL_MIDDLE_RIGHT	Label position middle right.
Datum.LABEL_NONE	No label.
Datum.LABEL_TOP_LEFT	Label position top left.
Datum.LABEL_TOP_RIGHT	Label position top right.
Datum.POINTS	Points type datum.

### Datum properties

Name	Type	Description
acronym	string	Datum acronym
fill_colour_above	<a href="#">Colour</a>	The colour above the datum line
fill_colour_below	<a href="#">Colour</a>	The colour below the datum line
fill_colour_left	<a href="#">Colour</a>	The colour left of the datum line
fill_colour_right	<a href="#">Colour</a>	The colour right of the datum line

fill_type	integer	The fill type. Can be <a href="#">Datum.FILL_ABOVE_BELOW</a> , <a href="#">Datum.FILL_RIGHT_LEFT</a> . Note that this can only be changed if the datum is of the type <a href="#">Datum.POINTS</a> .
label	string	Datum label
label_colour	<a href="#">Colour</a>	The colour of the datum label
label_position	integer	The label position. Can be <a href="#">Datum.LABEL_NONE</a> , <a href="#">Datum.LABEL_ABOVE_CENTRE</a> , <a href="#">Datum.LABEL_ABOVE_LEFT</a> , <a href="#">Datum.LABEL_ABOVE_RIGHT</a> , <a href="#">Datum.LABEL_BELOW_CENTRE</a> , <a href="#">Datum.LABEL_BELOW_LEFT</a> , <a href="#">Datum.LABEL_BELOW_RIGHT</a> , <a href="#">Datum.LABEL_MIDDLE_LEFT</a> , <a href="#">Datum.LABEL_TOP_LEFT</a> , <a href="#">Datum.LABEL_BOTTOM_LEFT</a> , <a href="#">Datum.LABEL_MIDDLE_RIGHT</a> , <a href="#">Datum.LABEL_TOP_RIGHT</a> , <a href="#">Datum.LABEL_BOTTOM_RIGHT</a> .
line_colour	<a href="#">Colour</a>	The colour of the datum line
line_style	<a href="#">LineStyle</a>	The line style used to draw the datum line
line_width	<a href="#">LineWidth</a>	The line width used to draw the datum line

## Detailed Description

The Datum class allows you to create and manipulate datums. See the documentation below for more details.

## Constructor

`new Datum(acronym[string], type[integer], value[real or array])`

## Description

Create a new [Datum](#) object. The datum will be added to all the currently active graphs.

## Arguments

Name	Type	Description
acronym	string	<a href="#">Datum</a> acronym
type	integer	Specify type of datum line. Can be <a href="#">Datum.CONSTANT_X</a> , <a href="#">Datum.CONSTANT_Y</a> , <a href="#">Datum.POINTS</a>
value	real or array	Value for <a href="#">Datum.CONSTANT_X</a> or <a href="#">Datum.CONSTANT_Y</a> type <a href="#">Datum</a> . If it is a <a href="#">Datum.POINTS</a> type <a href="#">Datum</a> then this should be an array of X, Y pairs.

## Return type

[Datum](#) object

## Example

To create a new datum with acronym my\_datum and a constant Y value of 100

```
var d = new Datum("my_datum", Datum.CONSTANT_Y, 100);
```

To create a new datum with acronym my\_datum and some X, Y points

```
var points = new Array(6);
points[0] = 0.0;
points[1] = 10.0;
points[2] = 1.0;
points[3] = 15.0;
points[4] = 2.0;
points[5] = 17.0;
var d = new Datum("my_datum", Datum.POINTS, points);
```

## Details of functions

### AddToGraph(graph, graph...*[int]*)

#### Description

Adds a datum to a graph.

#### Arguments

Name	Type	Description
graph, graph...	int	Optional list of graphs to add the datum to, If undefined then the datum is added to all graphs.

#### Return type

No return value.

#### Example

To add a datum (d) to graphs 1 and 3:

```
d.AddToGraph (1, 3) ;
```

To add a datum (d) to all graphs:

```
d.AddToGraph () ;
```

---

### Delete(datum*[string]*) [static]

#### Description

Deletes a datum

#### Arguments

Name	Type	Description
datum	string	Acronym of datum to delete

#### Return type

No return value

#### Example

To delete datum "my\_datum"

```
Datum.Delete ("my_datum") ;
```

---

### Exists(datum*[string]*) [static]

#### Description

Checks if a datum exists



## Arguments

Name	Type	Description
datum	string	Acronym of datum to check

## Return type

TRUE if the datum exists, otherwise FALSE

## Example

To check if a datum "my\_datum" exists

```
var exists = Datum.Exists("my_datum");
```

---

## First() [static]

## Description

Returns the first datum.

## Arguments

No arguments

## Return type

Datum object (or null if there are no datum in the model).

## Example

To get the 1st datum

```
var d = Datum.First();
```

---

## GetFromAcronym(datum[*string*]) [static]

## Description

Returns the datum object for a datum acronym.

## Arguments

Name	Type	Description
datum	string	Acronym of datum to return object for

## Return type

Datum object (or null if the datum does not exist).

## Example

To get the datum "my\_datum"

```
var d = Datum.GetFromAcronym("my_datum");
```

---

## Next()

### Description

Returns the next datum in the model.

### Arguments

No arguments

### Return type

Datum object (or null if there are no more datums in the model).

### Example

To get the next datum after datum d:  
`var datum = d.Next();`

---

## RemoveFromGraph(graph, graph...*[int]*)

### Description

Removes a datum from a graph.

### Arguments

Name	Type	Description
graph, graph...	int	Optional list of graphs to remove the datum from, If undefined then the datum is removed from all graphs.

### Return type

No return value.

### Example

To remove a datum (d) from graphs 1 and 3:  
`d.RemoveFromGraph(1,3);`

To remove a datum (d) from all graphs:  
`d.RemoveFromGraph();`

---

## Entity class

The Entity class contains constants relating to Entity types. [More...](#)

### Entity constants

Name	Description
Entity.AIRBAG	AIRBAG entity code
Entity.BEAM	BEAM entity code
Entity.BOUNDARY	BOUNDARY entity code
Entity.CONTACT	CONTACT entity code
Entity.FSI	FSI entity code
Entity.GEOMETRIC_CONTACT	GEOMETRIC CONTACT entity code
Entity.JOINT	JOINT entity code
Entity.MASS	MASS entity code
Entity.MODEL	MODEL entity code
Entity.NODAL_RB	NODAL RIGID BODY entity code
Entity.NODE	NODE entity code
Entity.NODE_GROUP	NODAL FORCE GROUP entity code
Entity.PART	PART entity code
Entity.PART_GROUP	PART GROUP entity code
Entity.PRETENSIONER	PRETENSIONER entity code
Entity.PULLEY	PULLEY entity code
Entity.RETRACTOR	RETRACTOR entity code
Entity.RIGIDWALL	RIGIDWALL entity code
Entity.SEATBELT	SEATBELT entity code
Entity.SHELL	SHELL entity code
Entity.SLIPRING	SLIPRING entity code
Entity.SOLID	SOLID entity code
Entity.SPC	SPC entity code
Entity.SPH	SPH entity code
Entity.SPRING	SPRING entity code
Entity.SUBSYSTEM	SUBSYSTEM entity code
Entity.THICK_SHELL	THICK SHELL entity code
Entity.TRACER	TRACER entity code
Entity.WELD	WELD entity code
Entity.X_SECTION	CROSS SECTION entity code

## Detailed Description

The Entity class is used to define entity type codes that can then be compared with the entity Curve property  
`Node = Entity.NODE;`

## File class

The File class allows you to read and write text files. [More...](#)

### Class functions

- [Copy](#)(source[*string*], dest[*string*])
- [Delete](#)(filename[*string*])
- [Exists](#)(filename[*string*])
- [FindFiles](#)(directory[*string*], type (optional)[*constant*])
- [Get](#)(url[*string*], filename[*string*], options (optional)[*object*])
- [IsAbsolute](#)(filename[*string*])
- [IsDirectory](#)(filename[*string*])
- [IsFile](#)(filename[*string*])
- [IsReadable](#)(filename[*string*])
- [IsWritable](#)(filename[*string*])
- [Mkdir](#)(directory[*string*])
- [Mktemp](#)()
- [Proxy](#)(name[*string*])
- [ProxyPassword](#)(name[*string*])
- [ProxyUsername](#)(username[*string*])
- [Rename](#)(oldname[*string*], newname[*string*])
- [Size](#)(filename[*string*])
- [Upload](#)(filename[*string*], url[*string*], options (optional)[*object*])

### Member functions

- [Close](#)()
- [FindLineContaining](#)(contain1[*string*], contain2 (optional)[*string*], contain3 (optional)[*string*], ... containn (optional)[*string*])
- [FindLineStarting](#)(start1[*string*], start2 (optional)[*string*], start3 (optional)[*string*], ... startn (optional)[*string*])
- [Flush](#)()
- [ReadArrayBuffer](#)(length (optional)[*integer*])
- [ReadChar](#)()
- [ReadLine](#)()
- [ReadLongLine](#)()
- [Seek](#)(offset[*integer*], origin (optional)[*constant*])
- [Tell](#)()
- [Write](#)(string[*Any valid javascript type*])
- [WriteArrayBuffer](#)(buffer[*ArrayBuffer*], length (optional)[*integer*])
- [Writeln](#)(string[*Any valid javascript type*])

### File constants

Name	Description
File.APPEND	Flag to open file for appending
File.BINARY	Flag to open file in binary mode. This will have no effect on unix/linux but for windows if a file is opened for writing with binary mode \n will not be translated to \r\n (CRLF), it will be written as \n (LF)
File.READ	Flag to open file for reading
File.UTF8	Flag to open file for reading as UTF-8 encoding.
File.WRITE	Flag to open file for writing

### Constants for Seek types

Name	Description
File.CURRENT	Seek relative to current file position
File.END	Seek relative to end of the file
File.START	Seek relative to start of the file

## Constants for Find types

Name	Description
File.DIRECTORY	Find directories
File.FILE	Find files

## File properties

Name	Type	Description
filename (read only)	string	Name of the file
mode (read only)	constant	Mode the file was opened with ( <a href="#">File.READ</a> , <a href="#">File.WRITE</a> etc)

## Detailed Description

The File class gives you simple functions to read and write text files. The following simple example shows how to read from the file "/data/test/file.txt" and print each line read to the dialogue box:

```
var f, line;
f = new File("/data/test/file.txt", File.READ);
while ( (line = f.ReadLine()) != undefined)
{
    Message(line);
}
f.Close();
```

The following simple example shows how to write the numbers 1 to 10 to the file "/data/test/file.txt":

```
var n, line;
f = new File("/data/test/file.txt", File.WRITE);
for (n=1; n<=10; n++)
{
    f.WriteLine(n);
}
f.Close();
```

See the documentation below for more details.

## Constructor

`new File(filename[string], mode[constant])`

## Description

Create a new [File](#) object for reading and writing text files.

## Arguments

Name	Type	Description
filename	string	Filename of the file you want to read/write. If reading, the file must exist. If writing, the file will be overwritten (if it exists) if mode is File.WRITE, or if mode is File.APPEND it will be appended to if it exists, or created if it does not. When reading a file the filename can also be a URL (uniform resource locator) in which case the file will be read from the remote site. See <a href="#">File.Get()</a> for more details on the format of the URL.
mode	constant	The mode to open the file with. Can be <a href="#">File.READ</a> , <a href="#">File.WRITE</a> or <a href="#">File.APPEND</a> . For <a href="#">File.WRITE</a> or <a href="#">File.APPEND</a> it can also be ORed with <a href="#">File.BINARY</a> if required. For <a href="#">File.READ</a> it can also be ORed with <a href="#">File.UTF8</a> if required.

## Return type

[File](#) object

## Example

To create a new file object to read file `"/data/test/file.txt"`

```
var f = new File("/data/test/file.txt", File.READ);
```

## Details of functions

### Close()

#### Description

Close a file opened by a [File](#) object.

#### Arguments

No arguments

#### Return type

No return value

#### Example

To close [File](#) object `f`.

```
f.Close();
```

---

### Copy(source[*string*], dest[*string*]) [static]

#### Description

Copies a file

#### Arguments

Name	Type	Description
source	string	Source filename you want to copy.
dest	string	Destination filename you want to copy source file to.

#### Return type

true if copy successful, error otherwise.

#### Example

To copy the file `"/data/test/file.key"` to `"/data/test/file.key_backup"`

```
var copied = File.Copy("/data/test/file.key", "/data/test/file.key_backup");
```

---

### Delete(filename[*string*]) [static]

#### Description

Deletes a file

## Arguments

Name	Type	Description
filename	string	Filename you want to delete.

## Return type

true if successful, false if not.

## Example

To delete the file "/data/test/file.key"  
`var deleted = File.Delete("/data/test/file.key");`

---

## Exists(filename[*string*]) [static]

## Description

Check if a file exists. See also [File.IsDirectory\(\)](#) and See also [File.IsFile\(\)](#).

## Arguments

Name	Type	Description
filename	string	Filename you want to check for existence.

## Return type

true/false

## Example

To see if the file "/data/test/file.key" exists  
`if (File.Exists("/data/test/file.key")) { do something }`

---

## FindFiles(directory[*string*], type (optional)[*constant*]) [static]

## Description

Find any files and/or directories in a directory.

## Arguments

Name	Type	Description
directory	string	Directory to look for files/directories in.
type (optional)	constant	Type of things to find. Can be bitwise OR of <a href="#">File.FILE</a> and <a href="#">File.DIRECTORY</a> . If omitted only files will be returned.

## Return type

Array of filenames/directories

---



## Example

To return the filenames in the directory /data/test:

```
var fileList = File.FindFiles("/data/test")
```

To return the directories in the directory /data/test:

```
var fileList = File.FindFiles("/data/test", File.DIRECTORY)
```

To return the files and directories in the directory /data/test:

```
var fileList = File.FindFiles("/data/test", File.FILE|File.DIRECTORY)
```

---

**FindLineContaining**(contain1[*string*], contain2 (optional)[*string*], contain3 (optional)[*string*], ... containn (optional)[*string*])

## Description

Reads a line from a file which contains **contain**, opened for reading by a [File](#) object. Although this is possible using core JavaScript functions this function should be significantly faster as most of the processing is done by T/HIS in C rather than in the JavaScript interpreter. To enable this function to be as fast as possible a maximum line length of 256 characters is used. If you expect a file to have lines longer than 256 characters then use [ReadLongLine](#) which allows lines of any length. If one argument is used then the line must contain that string. If more than one argument is used then lines which contain the string contain1 OR contain2 OR contain3 etc will be returned

## Arguments

Name	Type	Description
contain1	string	String which matching lines must contain (maximum length of 256 characters).
contain2 (optional)	string	alternative string which matching lines must contain (maximum length of 256 characters).
contain3 (optional)	string	alternative string which matching lines must contain (maximum length of 256 characters).
... containn (optional)	string	alternative string which matching lines must contain (maximum length of 256 characters).

## Return type

string read from file or undefined if end of file

## Example

Loop, reading lines from [File](#) object f which contain 'example'.

```
var line;
while ( (line = file.FindLineContaining("example") ) != undefined)
{
}
```

---

---

**FindLineStarting(start1[*string*], start2 (optional)[*string*], start3 (optional)[*string*], ... startn (optional)[*string*])**

## Description

Reads a line from a file which starts with start, opened for reading by a [File](#) object. Although this is possible using core JavaScript functions this function should be significantly faster as most of the processing is done by T/HIS in C rather than in the JavaScript interpreter. To enable this function to be as fast as possible a maximum line length of 256 characters is used. If you expect a file to have lines longer than 256 characters then use [ReadLongLine](#) which allows lines of any length. If one argument is used then the line must start with that string. If more than one argument is used then lines which start with start1 OR start2 OR start3 etc will be returned

## Arguments

Name	Type	Description
start1	string	String which matching lines must start with (maximum length of 256 characters).
start2 (optional)	string	alternative string which matching lines must start with (maximum length of 256 characters).
start3 (optional)	string	alternative string which matching lines must start with (maximum length of 256 characters).
... startn (optional)	string	alternative string which matching lines must start with (maximum length of 256 characters).

## Return type

string read from file or undefined if end of file

## Example

Loop, reading lines from [File](#) object f which start 'example'.

```
var line;
while ( (line = file.FindLineStarting("example") ) != undefined)
{
}
```

---

## Flush()

## Description

Flushes a file opened for writing by a [File](#) object.

## Arguments

No arguments

## Return type

No return value

## Example

To flush [File](#) object f.

```
f.Flush();
```

---

---

**Get(url[*string*], filename[*string*], options (optional)[*object*]) [static]****Description**

Get a file from a remote location. See also [File.Proxy\(\)](#), [File.ProxyPassword\(\)](#) and [File.ProxyUsername\(\)](#).

**Arguments**

Name	Type	Description
url	string	URL (uniform resource locator) of remote file you want to get. Currently http and ftp are supported. For http give the full address including the leading 'http://'. e.g. 'http://www.example.com/file.html'. For ftp an optional username and password can be given. e.g. 'ftp://ftp.example.com' retrieves the directory listing for the root directory. 'ftp://ftp.example.com/readme.txt' downloads the file readme.txt from the root directory. 'ftp://user:password@ftp.example.com/readme.txt' retrieves the readme.txt file from the user's home directory.
filename	string	Filename you want to save the file to.
options (optional)	object	Options for get. Currently the only available properties are 'username' (string), 'password' (string) and 'response' (boolean). If 'username' and 'password' are set then basic authorization using the username and password will be used. If 'response' is used and is true then the response code will be returned instead of true/false. This can be used to retrieve error messages and codes when the file is not returned successfully.

**Return type**

true if file was successfully got, false otherwise.

**Example**

To get the file "http://www.example.com/file.html" and save it to C:\temp:  

```
File.Get("http://www.example.com/file.html", "C:\temp\file.html");
```

---

**IsAbsolute(filename[*string*]) [static]****Description**

Check if a filename is absolute or relative.

**Arguments**

Name	Type	Description
filename	string	Filename you want to check.

**Return type**

true/false

**Example**

To see if the filename "/data/test" is absolute (which it is!)  

```
if (File.IsAbsolute("/data/test")) { do something }
```

## IsDirectory(filename[*string*]) [static]

### Description

Check if a filename is a directory. See also [File.Exists\(\)](#), [File.IsFile\(\)](#), [File.IsReadable\(\)](#) and [File.IsWritable\(\)](#).

### Arguments

Name	Type	Description
filename	string	Filename you want to check.

### Return type

true/false

### Example

To see if the filename "/data/test" is a directory  

```
if (File.IsDirectory("/data/test")) { do something }
```

---

## IsFile(filename[*string*]) [static]

### Description

Check if a filename is a file. See also [File.Exists\(\)](#), [File.IsDirectory\(\)](#), [File.IsReadable\(\)](#) and [File.IsWritable\(\)](#).

### Arguments

Name	Type	Description
filename	string	Filename you want to check.

### Return type

true/false

### Example

To see if the filename "/data/test" is a file  

```
if (File.IsFile("/data/test")) { do something }
```

---

## IsReadable(filename[*string*]) [static]

### Description

Check if a filename has read permissions. See also [File.Exists\(\)](#), [File.IsDirectory\(\)](#) and [File.IsWritable\(\)](#).

### Arguments

Name	Type	Description
filename	string	Filename you want to check.

### Return type

true/false

---

## Example

To see if the filename "/data/test" is writable  

```
if (File.IsWritable("/data/test")) { do something }
```

---

## IsWritable(filename[*string*]) [static]

### Description

Check if a filename has write permissions. See also [File.Exists\(\)](#), [File.IsDirectory\(\)](#) and [File.IsReadable\(\)](#).

### Arguments

Name	Type	Description
filename	string	Filename you want to check.

### Return type

true/false

### Example

To see if the filename "/data/test" is writable  

```
if (File.IsWritable("/data/test")) { do something }
```

---

## Mkdir(directory[*string*]) [static]

### Description

Make a directory.

### Arguments

Name	Type	Description
directory	string	The name of the directory you want to create.

### Return type

true if successfully created, false if not.

### Example

To make the directory "/data/test"  

```
var success = File.Mkdir("/data/test");
```

---

## Mktemp() [static]

### Description

Make a temporary filename for writing a temporary file.

---

## Arguments

No arguments

### Return type

String name of temporary filename that can be used.

### Example

```
To get a temp filename"
var filename = File.Mktemp();
```

---

## Proxy(name[*string*]) [static]

### Description

Set a proxy for files opened by http, ftp etc. See also [File.Get\(\)](#), [File.ProxyPassword\(\)](#) and [File.ProxyUsername\(\)](#).

### Arguments

Name	Type	Description
name	string	The name of the proxy.

### Return type

No return value

### Example

```
To set the proxy to "http://example.proxy.com" using port 80:
File.Proxy("http://example.proxy.com:80");
```

---

## ProxyPassword(name[*string*]) [static]

### Description

Set a proxy password for files opened by http, ftp etc. See also [File.Get\(\)](#), [File.Proxy\(\)](#) and [File.ProxyUsername\(\)](#).

### Arguments

Name	Type	Description
name	string	Password for the proxy server.

### Return type

No return value

### Example

```
To set the proxy password to "password":
File.ProxyPassword("password");
```

---

---

## ProxyUsername(username[*string*]) [static]

### Description

Set a proxy username for files opened by http, ftp etc. See also [File.Get\(\)](#), [File.Proxy\(\)](#) and [File.ProxyPassword\(\)](#).

### Arguments

Name	Type	Description
username	string	The username for the proxy.

### Return type

No return value

### Example

To set the proxy username to "username":  
`File.ProxyUsername("username");`

---

## ReadArrayBuffer(length (optional)[*integer*])

### Description

Reads binary data from a file opened for reading by a [File](#) object. The data is returned as an [ArrayBuffer](#) object. For more details on how to use an [ArrayBuffer](#) see the following links:

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays](https://developer.mozilla.org/en/JavaScript_typed_arrays)

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays/ArrayBuffer](https://developer.mozilla.org/en/JavaScript_typed_arrays/ArrayBuffer)

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays/ArrayBufferView](https://developer.mozilla.org/en/JavaScript_typed_arrays/ArrayBufferView)

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays/DataView](https://developer.mozilla.org/en/JavaScript_typed_arrays/DataView).

### Arguments

Name	Type	Description
length (optional)	integer	Number of bytes to try to read from the file. If omitted all the remaining data from the file will be read.

### Return type

[ArrayBuffer](#) object or undefined if end of file

### Example

To read data as 32bit unsigned integers from [File](#) object f.

```

var ab = f.ReadArrayBuffer();
var u32 = new Uint32Array(ab);
for (var i=0; i<u32.length; i++)
{
    var value = u32[i];
}

```

## ReadChar()

### Description

Reads a single character from a file opened for reading by a [File](#) object.

### Arguments

No arguments

### Return type

character read from file or  
undefined

if end of file

### Example

Loop, reading characters from [File](#) object f.

```
var c;  
while ( (c = f.ReadChar()) != undefined) { ... }
```

---

## ReadLine()

### Description

Reads a line from a file opened for reading by a [File](#) object. To enable this function to be as fast as possible a maximum line length of 256 characters is used. If you expect a file to have lines longer than 256 characters then use [ReadLongLine](#) which allows lines of any length.

### Arguments

No arguments

### Return type

string read from file or  
undefined

if end of file

### Example

Loop, reading lines from [File](#) object f.

```
var line;  
while ( (line = f.ReadLine()) != undefined) { ... }
```

---

## ReadLongLine()

### Description

Reads a line from a file opened for reading by a [File](#) object. The line can be any length. If your file has lines shorter than 256 characters then you may want to use [ReadLine](#) instead which is faster.



## Arguments

No arguments

### Return type

string read from file or  
undefined

if end of file

### Example

Loop, reading lines from [File](#) object f.

```
var line;
while ( (line = f.ReadLongLine()) != undefined) { ... }
```

---

## Rename(oldname[*string*], newname[*string*]) [static]

### Description

Rename an existing file to have a different name.

### Arguments

Name	Type	Description
oldname	string	Existing filename you want to rename
newname	string	New filename you want to rename to

### Return type

true if successful, false if not.

### Example

To get the size of the file "/data/test/file.key"

```
var size = File.Size("/data/test/file.key");
```

---

## Seek(offset[*integer*], origin (optional)[*constant*])

### Description

Set the current position for reading or writing in a [File](#) object.

### Arguments

Name	Type	Description
offset	integer	Offset to seek to in the file
origin (optional)	constant	Origin for offset. Must be one of <a href="#">File.START</a> , <a href="#">File.END</a> or <a href="#">File.CURRENT</a> . If omitted <a href="#">File.START</a> will be used.

### Return type

no return value

## Example

To seek to the end of [File](#) f:  
`f.Seek(0, File.END);`

To seek to the beginning of [File](#) f:  
`f.Seek(0, File.START);`

To move forward 10 characters in [File](#) f:  
`f.Seek(10, File.CURRENT);`

---

## Size(filename[*string*]) [static]

### Description

Return the size of a file in bytes

### Arguments

Name	Type	Description
filename	string	Filename you want the size of.

### Return type

size in bytes

### Example

To get the size of the file `"/data/test/file.key"`  
`var size = File.Size("/data/test/file.key");`

---

## Tell()

### Description

Return the current file position for a [File](#) object. Note that on Windows when reading files if the file is not opened with [File.BINARY](#) this may not return the correct file position for files with unix line endings.

### Arguments

No arguments

### Return type

integer

### Example

To get the current file position for [File](#) f:  
`var pos = f.Tell();`

---

---

Upload(filename[*string*], url[*string*], options (optional)[*object*]) [static]

## Description

Uploads a file to a remote location. See also [File.Proxy\(\)](#), [File.ProxyPassword\(\)](#) and [File.ProxyUsername\(\)](#).

## Arguments

Name	Type	Description
filename	string	Filename you want to upload.
url	string	URL (uniform resource locator) of the remote location you want to upload the file to. Currently only http is supported. Give the full address including the leading 'http://'. e.g. 'http://www.example.com/file.html'.
options (optional)	object	Options for upload. Currently the only available properties are 'username' and 'password'. If both of these are set then basic authorization using the username and password will be used.

## Return type

true if file was successfully uploaded, false otherwise.

## Example

To upload the file "C:\temp\file.txt" to "http://www.example.com/file.txt":  

```
File.Upload("C:/temp/file.txt", "http://www.example.com/file.txt");
```

---

Write(string[*Any valid javascript type*])

## Description

Write a string to a file opened for writing by a [File](#) object. **Note that a carriage return is not added.**

## Arguments

Name	Type	Description
string	Any valid javascript type	The string/item that you want to write

## Return type

No return value

## Example

To write string "Hello, world!" to [File](#) object f  

```
f.Write("Hello, world!\n");
```

To write the title of model m to [File](#) object f  

```
f.Write("The title of model 2 is " + m.title + "\n");
```

---

---

**WriteArrayBuffer(buffer[[ArrayBuffer](#)], length (optional)[*integer*])**

## Description

Writes binary data to a file opened for writing by a [File](#) object. The data to write is an [ArrayBuffer](#) object. For more details on how to use an [ArrayBuffer](#) see the following links:

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays](https://developer.mozilla.org/en/JavaScript_typed_arrays)

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays/ArrayBuffer](https://developer.mozilla.org/en/JavaScript_typed_arrays/ArrayBuffer)

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays/ArrayBufferView](https://developer.mozilla.org/en/JavaScript_typed_arrays/ArrayBufferView)

[https://developer.mozilla.org/en/JavaScript\\_typed\\_arrays/DataView](https://developer.mozilla.org/en/JavaScript_typed_arrays/DataView).

## Arguments

Name	Type	Description
buffer	<a href="#">ArrayBuffer</a>	<a href="#">ArrayBuffer</a> to write to file
length (optional)	integer	Number of bytes to write to the file. If omitted all the data in the <a href="#">ArrayBuffer</a> will be written (buffer.byteLength bytes)

## Return type

No return value

## Example

To write [ArrayBuffer](#) ab to [File](#) object f.

```
f.WriteArrayBuffer(ab);
```

---

**WriteLn(string[*Any valid javascript type*])**

## Description

Write a string to a file opened for writing by a [File](#) object **adding a carriage return**.

## Arguments

Name	Type	Description
string	Any valid javascript type	The string/item that you want to write

## Return type

No return value

## Example

To write string "Hello, world!" to [File](#) object f automatically adding a carriage return

```
f.WriteLine("Hello, world!");
```

To write the title of model m to [File](#) object f automatically adding a carriage return

```
f.WriteLine("The title of model 2 is " + m.title);
```

---

## Graph class

The Graph class gives you access to graphs in T/HIS. [More...](#)

### Class functions

- [Total\(\)](#)

### Graph properties

Name	Type	Description
id	integer	Graph ID (read only)

### Detailed Description

The Graph class contains information on the number of graphs. See the documentation below for more details.

### Constructor

```
new Graph(index[integer])
```

### Description

Create a new [Graph](#).

### Arguments

Name	Type	Description
index	integer	Graph index to copy initial display and axis settings from (optional). If not defined then the display and axis settings will be copied from those defined in the preference file.

### Return type

[Graph](#) object

### Example

To create a new graph and copy all of the setting from graph 2

```
var l = new Graph(2);
```

### Details of functions

#### Total() [static]

### Description

Returns the total number of graphs.

### Arguments

No arguments

### Return type

integer

## Example

To find how many graphs there are in T/HIS:  
`var num = Graph.Total();`

---

## Group class

The Group class gives you access to groups in T/HIS. [More...](#)

### Class functions

- [Get](#)(Name[*string*])
- [GetFromID](#)(ID[*integer*])
- [Total](#)()

### Member functions

- [Add](#)(Curve[*Curve*])
- [AddAll](#)()
- [AddID](#)(ID[*integer*])
- [Contains](#)(Curve[*Curve*])
- [ContainsID](#)(ID[*integer*])
- [GetCurveIDs](#)()
- [GetCurves](#)()
- [Remove](#)(Curve[*Curve*])
- [RemoveAll](#)()
- [RemoveID](#)(ID[*integer*])
- [Spool](#)()
- [SpoolID](#)()
- [StartSpool](#)()

### Group properties

Name	Type	Description
curves	integer	Number of curves in the group (read only)
name	string	Group name (read only)

### Detailed Description

The Group class allows you to create, and modify groups. See the documentation below for more details.

### Constructor

`new Group(name[string])`

### Description

Create a new [Group](#) object.

### Arguments

Name	Type	Description
name	string	Group name used to reference the group

### Return type

[Group](#) object

### Example

To create a new group with the name X-Velocity

```
var l = new Group("X-velocity");
```

## Details of functions

### Add(Curve[[Curve](#)])

#### Description

Adds a curve object to group.

#### Arguments

Name	Type	Description
Curve	<a href="#">Curve</a>	<a href="#">Curve</a> that will be added to group

#### Return type

No return value.

#### Example

To add curve c to curve group g:  
`g.Add (c) ;`

---

### AddAll()

#### Description

Adds all curves to group.

#### Arguments

No arguments

#### Return type

No return value.

#### Example

To add all curves to curve group g:  
`g.AddAll () ;`

---

### AddID(ID[*integer*])

#### Description

Adds curve by ID to a group.

#### Arguments

Name	Type	Description
ID	integer	The ID of the curve you want to add.



---

## Return type

No return value.

### Example

To add curve 3 to curve group g:

```
g.AddID(3);
```

---

## Contains(Curve[Curve])

### Description

Checks if a curve object is in a curve group.

### Arguments

Name	Type	Description
Curve	Curve	Curve that will be checked

### Return type

TRUE if the curve is in the group, otherwise FALSE

### Example

To check if a curve object n is in group g

```
var exists = g.Contains(n);
```

---

## ContainsID(ID[integer])

### Description

Checks if a curve ID is in a curve group.

### Arguments

Name	Type	Description
ID	integer	The ID of the curve you want to check.

### Return type

TRUE if the curve is in the group, otherwise FALSE

### Example

To check if a curve ID n is in group g

```
var exists = g.ContainsID(n);
```

---

## Get(Name[*string*]) [static]

### Description

Returns a group object.

### Arguments

Name	Type	Description
Name	string	Name of the group to return object for

### Return type

Group object (or Null if the group does not exist).

### Example

To get the group called 'left'  
`var group = Group.Get("left");`

---

## GetCurveIDs()

### Description

Returns an array of Curve ID's for all the Curves in the group.

### Arguments

No arguments

### Return type

Array of Curve ID's.

### Example

To make an array of Curve ID's for all the curves in group g:  
`var curves = g.GetCurveIDs();`

---

## GetCurves()

### Description

Returns an array of Curve Objects for all the Curves in the group.

### Arguments

No arguments

### Return type

Array of Curve objects.

---

## Example

To make an array of Curve objects for all the curves in group g:

```
var curves = g.GetCurves();
```

---

## GetFromID(ID[integer]) [static]

### Description

Returns a group object.

### Arguments

Name	Type	Description
ID	integer	ID of the group to return object for

### Return type

Group object (or Null if the group does not exist).

### Example

To get the group number 1

```
var group = Group.GetFromID(1);
```

---

## Remove(Curve[Curve])

### Description

Removes a curve object from a group.

### Arguments

Name	Type	Description
Curve	<a href="#">Curve</a>	<a href="#">Curve</a> that will be removed from group

### Return type

No return value.

### Example

To remove curve c from curve group g:

```
g.Remove(c);
```

---

## RemoveAll()

### Description

Removes all curves from a group.

## Arguments

No arguments

## Return type

No return value.

## Example

To remove all curves from curve group g:  
`g.RemoveAll () ;`

---

## RemoveID(ID[integer])

## Description

Remove a curve by from a group.

## Arguments

Name	Type	Description
ID	integer	The ID of the curve you want to remove.

## Return type

No return value.

## Example

To remove curve 3 from curve group g:  
`g.RemoveID (3) ;`

---

## Spool()

## Description

Spools a group, entry by entry and returns the curve objects. See also [Group.StartSpool](#)

## Arguments

No arguments

## Return type

Curve Object of item, or NULL if no more curves in group

## Example

```
To spool group g:
var id;
g.StartSpool ();
while (id = g.Spool () )
{
    do something...
}
```

---

## SpoolID()

### Description

Spools a group, entry by entry and returns the curve ID's. See also [Group.StartSpool](#)

### Arguments

No arguments

### Return type

Curve ID, or 0 if no more curves in group

### Example

```
To spool group g :  
var id;  
g.StartSpool();  
while (id = g.SpoolID() )  
{  
    do something...  
}
```

---

## StartSpool()

### Description

Starts a group spooling operation. See also [Group.Spool](#)

### Arguments

No arguments

### Return type

No return value

### Example

```
To start spooling group g:  
g.StartSpool();
```

---

## Total() [static]

### Description

Returns the total number of curve group currently defined

### Arguments

No arguments

### Return type

Number of curve groups currently defined.

---

## Example

To get the number of curve groups

```
var total = Group.Total();
```

---

## LineStyle class

The LineStyle class contains constants relating to the curve line style. [More...](#)

### LineStyle constants

Name	Description
LineStyle.DASH	Dashes lines
LineStyle.DASH2	Dash pattern 2
LineStyle.DASH3	Dash pattern 3
LineStyle.DASH4	Dash pattern 4
LineStyle.DASH5	Dash pattern 5
LineStyle.DASH6	Dash pattern 6
LineStyle.NONE	No line
LineStyle.SOLID	Solid lines

### Detailed Description

The LineStyle class is used to define the line style used to draw curves:  
`p.style = LineStyle.SOLID;`

## LineWidth class

The LineWidth class contains constants relating to the curve line width. [More...](#)

### LineWidth constants

Name	Description
LineWidth.BOLD	Bold lines (4 pixels wide)
LineWidth.FINE	Fine lines (1 pixel wide)
LineWidth.HEAVY	Heavy lines (8 pixels wide)
LineWidth.NORMAL	Normal lines (2 pixels wide)

### Detailed Description

The LineWidth class is used to define the line width used to draw curves:  
`p.width = LineWidth.NORMAL;`



## Model class

The Model class gives you access to models in T/HIS. [More...](#)

### Class functions

- [Exists](#)(model number[integer])
- [GetFromID](#)(model number[integer])
- [HighestID](#)()
- [Total](#)()

### Model properties

Name	Type	Description
dir	string	Directory containing the model file (read only).
file	string	File selected when reading the model (read only).
id	integer	Model ID (read only)
title	string	Model title (read only).

### Detailed Description

The Model class contains information on filenames and directories belonging to a model. See the documentation below for more details.

### Details of functions

#### Exists(model number[integer]) [static]

##### Description

Checks if a model exists

##### Arguments

Name	Type	Description
model number	integer	number of the model you want to check the existence of

##### Return type

TRUE if the model exists, otherwise FALSE

##### Example

To check if a model n exists

```
var exists = Model.Exists(n);
```

---

#### GetFromID(model number[integer]) [static]

##### Description

Returns the Model object for a model ID or null if model does not exist.

## Arguments

Name	Type	Description
model number	integer	number of the model you want the Model object for

## Return type

Model object (or null if model does not exist).

## Example

To get the model n  
`var model = Model.GetFromID(n);`

---

## HighestID() [static]

### Description

Returns the ID of the highest model currently being used

### Arguments

No arguments

### Return type

ID of highest model currently being used.

### Example

To get the highest model ID  
`var id= Model.HighestID();`

---

## Total() [static]

### Description

Returns the total number of models.

### Arguments

No arguments

### Return type

integer

### Example

To find how many models there are in T/HIS:  
`var num = Model.Total();`

---

## Operate class

The Operate class gives you access to the built in curve operations in T/HIS. [More...](#)

### Class functions

- [Abs](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Acos](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Acu](#)(Input Curve[[Curve](#)], Offset[[float](#)], Time Period[[float](#)], Output Curve (optional)[[Curve](#)])
- [Ad](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Add](#)(Input Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Adx](#)(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Asi](#)(X Acceleration[[Curve](#)], Y Acceleration[[Curve](#)], Z Acceleration[[Curve](#)], Acceleration conversion factor[[float](#)], X Acceleration Limit[[float](#)], Y Acceleration Limit[[float](#)], Z Acceleration Limit[[float](#)], Calculation method[[string](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)])
- [Asin](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Atan](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Ave](#)(Curve array[[array](#)], Output Curve (optional)[[Curve](#)])
- [Blc](#)(Input Curve[[Curve](#)])
- [But](#)(Input Curve[[Curve](#)], Frequency[[float](#)], Order[[integer](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)])
- [C1000](#)(Input Curve[[Curve](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)])
- [C180](#)(Input Curve[[Curve](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)])
- [C60](#)(Input Curve[[Curve](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)])
- [C600](#)(Input Curve[[Curve](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)])
- [Cat](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Clip](#)(Input Curve[[Curve](#)], X min[[float](#)], X max[[float](#)], Y min[[float](#)], Y max[[float](#)], Output Curve (optional)[[Curve](#)])
- [Com](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Cor](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)], Correlation type[[string](#)])
- [Cor3](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)], X axis factor (optional)[[float](#)], Y axis factor (optional)[[float](#)])
- [Cos](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Da](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Dif](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Div](#)(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Dix](#)(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Ds](#)(Input Curve[[Curve](#)], Broadening Factor[[float](#)], Redefine Frequencies[[string](#)], Output Curve (optional)[[Curve](#)])
- [Dv](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Env](#)(Curve array[[array](#)], Output Curve (optional)[[Curve](#)])
- [Err](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Exc](#)(Input Curve[[Curve](#)], Output option[[string](#)], Output Curve (optional)[[Curve](#)])
- [Exp](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Fft](#)(Input Curve[[Curve](#)], Output option[[string](#)], X axis interval (optional)[[float](#)], Output option[[string](#)])
- [Fir](#)(Input Curve[[Curve](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)])
- [Hic](#)(Input Curve[[Curve](#)], Window[[float](#)], Acceleration factor[[float](#)])
- [Hicd](#)(Input Curve[[Curve](#)], Window[[float](#)], Acceleration factor[[float](#)])
- [Ifft](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)], Input type[[string](#)])
- [Int](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Log](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Log10](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Log10x](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Logx](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Lsq](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Map](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Max](#)(Curve array[[array](#)], Output Curve (optional)[[Curve](#)])
- [Min](#)(Curve array[[array](#)], Output Curve (optional)[[Curve](#)])
- [Mon](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Mul](#)(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Mux](#)(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or [real](#)], Output Curve (optional)[[Curve](#)])
- [Ncp](#)(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)])
- [Nij](#)(Shear Force[[Curve](#)], Axial Force[[Curve](#)], Moment[[Curve](#)], Fzc(tension)[[float](#)], Fzc(compression)[[float](#)], Myc(Flexion)[[float](#)], Myc(Extension)[[float](#)], E[[float](#)])
- [Nor](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Nox](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Octave](#)(Input Curve[[Curve](#)], Band type to convert to[[String](#)], Output Type[[String](#)], Output Type[[String](#)], Output Curve (optional)[[Curve](#)])
- [Order](#)(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)])
- [Pbut](#)(Input Curve[[Curve](#)], Frequency[[float](#)], Order[[integer](#)], X axis interval (optional)[[float](#)], Output Curve

- (optional)/[Curve](#))
- [Power](#)(Input Curve/[Curve](#), Power/[float](#), Output Curve (optional)/[Curve](#))
- [Rave](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Rec](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Reg](#)(Input Curve/[Curve](#), X axis interval/[float](#), Output Curve (optional)/[Curve](#))
- [Res](#)(Curve array/[array](#), Output Curve (optional)/[Curve](#))
- [Rev](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Rs](#)(Input Curve/[Curve](#), Damping Factor/[float](#), Sampling Points/[int](#), X axis interval (optional)/[float](#), Output Curve (optional)/[Curve](#))
- [Sin](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Smooth](#)(Input Curve/[Curve](#), Smoothing Factor/[integer](#), Output Curve (optional)/[Curve](#))
- [Sqr](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Stress](#)(Input Curve/[Curve](#), Convert to/[string](#), Output Curve (optional)/[Curve](#))
- [Sub](#)(1st Curve/[Curve](#), 2nd Curve or constant/[Curve or real](#), Output Curve (optional)/[Curve](#))
- [Sum](#)(Curve array/[array](#), Output Curve (optional)/[Curve](#))
- [Sux](#)(1st Curve/[Curve](#), 2nd Curve or constant/[Curve or real](#), Output Curve (optional)/[Curve](#))
- [Tan](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Thiv](#)(X Acceleration/[Curve](#), Y Acceleration/[Curve](#), Yaw Rate/[Curve](#), Dx/[float](#), Dy/[float](#), X0/[float](#))
- [Tms](#)(Input Curve/[Curve](#), Period/[float](#))
- [Translate](#)(Input Curve/[Curve](#), X value/[float](#), Y value/[float](#), Output Curve (optional)/[Curve](#))
- [Tti](#)(Upper Rib Acceleration/[Curve](#), Lower Rib Acceleration/[Curve](#), T12 Acceleration/[Curve](#))
- [Va](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#), Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Vc](#)(Input Curve/[Curve](#), A/[float](#), B/[float](#), Calculation method/[string](#), Output Curve (optional)/[Curve](#))
- [Vd](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [Vec](#)(1st Curve/[Curve](#), 2nd Curve/[Curve or real](#), 3rd Curve/[Curve or real](#), Output Curve (optional)/[Curve](#))
- [Vec2d](#)(1st Curve/[Curve](#), 2nd Curve/[Curve or real](#), Output Curve (optional)/[Curve](#))
- [Wif](#)(1st Curve/[Curve](#), 2nd Curve/[Curve](#))
- [Window](#)(Input Curve/[Curve](#), Window Type/[string](#), %age lead in (optional - cosine only)/[float](#), Output Curve (optional)/[Curve](#))
- [Zero](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [ZeroX](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [ZeroY](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#))
- [dB](#)(Input Curve/[Curve](#), Reference Value/[float](#), Output Curve (optional)/[Curve](#))
- [dBA](#)(Input Curve/[Curve](#), Weighting Type/[String](#), Output Curve (optional)/[Curve](#))

## Detailed Description

The Operate class allows you to use the built in curve operations in T/HIS to generate new curves. Most of the curve operations generate a new curve and return the curve object for the new curve. A few functions (NIJ, FFT, etc) generate multiple output curves and these return an array of curve objects.

See the documentation below for more details.

## Details of functions

[Abs](#)(Input Curve/[Curve](#), Output Curve (optional)/[Curve](#)) [static]

### Description

Convert a curve to absolute values

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

## Example

To convert curve m to absolute values and store as curve p  
`p = Operate.Abs (m) ;`

---

**Acos**(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Arc Cosine

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Calculate Arc Cosine() of curve m and store as curve p  
`p = Operate.Acos (m) ;`

---

**Acu**(Input Curve[[Curve](#)], Offset[*float*], Time Period[*float*], Output Curve (optional)[[Curve](#)]) [static]

## Description

Evaluates the integratal of a curve over a user defined period

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Offset	float	User defined offset
Time Period	float	Time to integrate over
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Integrate c curve over 0.07 seconds with a 0.1 offset.  
`p = Operate.Acu (m, 0.1, 0.007) ;`

---

---

Ad(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

### Description

Convert acceleration spectrum to a displacement spectrum

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Convert curve m and store as curve p

```
p = Operate.Ad (m) ;
```

---

Add(Input Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or real], Output Curve (optional)[[Curve](#)]) [static]

### Description

Add Y axis values

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

To add curves m and n together and store as curve p

```
p = Operate.Add (m,n) ;
```

To add 20.0 to the values in curve m and store as curve p

```
p = Operate.Add (m, 20.0) ;
```

---

Adx(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or real], Output Curve (optional)[[Curve](#)]) [static]

### Description

Add X axis values

---

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To add X axis values for curves m and n together and store as curve p  
`p = Operate.Adx (m, n) ;`

To add 20.0 to the X axis values in curve m and store as curve p  
`p = Operate.Adx (m, 20.0) ;`

Asi(X Acceleration[[Curve](#)], Y Acceleration[[Curve](#)], Z Acceleration[[Curve](#)], Acceleration conversion factor[*float*], X Acceleration Limit[*float*], Y Acceleration Limit[*float*], Z Acceleration Limit[*float*], Calculation method[*string*], X axis interval (optional)[*float*], Output Curve (optional)[[Curve](#)] [static]

## Description

Acceleration Severity Index. This value is used to assess the performance of road side crash barriers. The calculation method can be set to 2010 (BS EN 1317-1:2010) or 1998 (BS EN 1317-1:1998).

## Arguments

Name	Type	Description
X Acceleration	<a href="#">Curve</a>	X Acceleration <a href="#">Curve</a>
Y Acceleration	<a href="#">Curve</a>	Y Acceleration <a href="#">Curve</a>
Z Acceleration	<a href="#">Curve</a>	Z Acceleration <a href="#">Curve</a>
Acceleration conversion factor	float	Factor required to divide input acceleration curve by to convert to (G)
X Acceleration Limit	float	X direction acceleration limit
Y Acceleration Limit	float	Y direction acceleration limit
Z Acceleration Limit	float	Z direction acceleration limit
Calculation method	string	Either 2010 or 1998.
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate ASI using the 2010 method with input curves x,y and z, factors 12,9,10 and a conversion factor of 9810. Regularise the input curves using an interval of 0.0001 first.

```
p = Operate.Asi(x,y,z,9810.0,12.0,9.0,10.0,"2010",0.0001);
```

---

Asin(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Arc Sine

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

## Example

Calculate Arc Sine() of curve m and store as curve p

```
p = Operate.Asin(m);
```

---

Atan(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Arc Tangent

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

## Example

Calculate Arc Tangent() of curve m and store as curve p

```
p = Operate.Atan(m);
```



---

Ave(Curve array[*array*], Output Curve (optional)[*Curve*]) [static]

### Description

Average a group of curves

### Arguments

Name	Type	Description
Curve array	array	Array of <a href="#">Curve</a> objects
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Average the array of curves stored in curve array x and store as curve p  
 p = Operate.Ave(x);

---

Blc(Input Curve[*Curve*]) [static]

### Description

Carry out a baseline correction on an acceleration time history

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Moment / Time <a href="#">Curve</a>

### Return type

Array of [Curve](#) objects. 1st curve : Corrected curve 2nd curve : Integrated Velocity 3rd curve : Integrated Displacement

### Example

Calculate baseline correction on curve m, .  
 c\_array = Operate.Blc(m);  
 corrected\_curve = c\_array[0];  
 vel\_curve = c\_array[1];  
 disp\_curve = c\_array[2];

---

But(Input Curve[*Curve*], Frequency[*float*], Order[*integer*], X axis interval (optional)[*float*],  
 Output Curve (optional)[*Curve*]) [static]

### Description

Butterworth Filter

---

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Frequency	float	Cut-off Frequency (Hz)
Order	integer	Filter order
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Filter curve m using a cut-off of 400Hz and order 2 and output as curve p . Regularise the input curve using an interval of 0.0001 first.

```
p = Operate.But (m, 400.0, 2, 0.0001) ;
```

---

C1000(Input Curve[\[Curve\]](#), X axis interval (optional)[\[float\]](#), Output Curve (optional)[\[Curve\]](#))  
[static]

## Description

SAE Class 1000 Filter

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Filter curve m and output as curve p . Regularise the input curve using an interval of 0.0001 first.

```
p = Operate.C1000 (m, 0.0001) ;
```

---

C180(Input Curve[\[Curve\]](#), X axis interval (optional)[\[float\]](#), Output Curve (optional)[\[Curve\]](#))  
[static]

## Description

SAE Class 180 Filter

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Filter curve m and output as curve p . Regularise the input curve using an interval of 0.0001 first.  
 p = Operate.C180(m, 0.0001) ;

---

C60(Input Curve[[Curve](#)], X axis interval (optional)[float], Output Curve (optional)[[Curve](#)])  
 [static]

## Description

SAE Class 60 Filter

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Filter curve m and output as curve p . Regularise the input curve using an interval of 0.0001 first.  
 p = Operate.C60(m, 0.0001) ;

---

C600(Input Curve[[Curve](#)], X axis interval (optional)[float], Output Curve (optional)[[Curve](#)])  
 [static]

## Description

SAE Class 600 Filter

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Filter curve m and output as curve p . Regularise the input curve using an interval of 0.0001 first.  
`p = Operate.C600(m, 0.0001) ;`

---

Cat(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or real], Output Curve (optional)[[Curve](#)]) [static]

## Description

Concatenate 2 curves together

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To concatenate the values for curve n to those in curve m and store as curve p  
`p = Operate.Cat(m, n) ;`

---

Clip(Input Curve[[Curve](#)], X min[*float*], X max[*float*], Y min[*float*], Y max[*float*], Output Curve (optional)[[Curve](#)]) [static]

## Description

Clip a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X min	float	X minimum value
X max	float	X maximum value
Y min	float	Y minimum value
Y max	float	Y maximum value
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Clip a curve m to within  $0.1 < x < 0.3$ ,  $0.0 < y < 100.0$  and store as curve p  
`p = Operate.Clip(m, 0.1, 0.3, 0.0, 100.0) ;`

---

Com(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or real], Output Curve (optional)[[Curve](#)]) [static]

### Description

Combine Y axis values from 2 curves together

### Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

To combine the Y axis values for curve n to those in curve m and store as curve p  
`p = Operate.Com(m, n) ;`

---

Cor(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)], Correlation type[*string*]) [static]

### Description

Curve Correlation function. This Correlation function provides a measure of the degree to which two curves match. When comparing curves by eye, the quality of correlation may be judged on the basis of how well matched are the patterns of peaks, the overall shapes of the curves, etc, and can allow for differences of timing as well as magnitude. Thus a simple function based on the difference of Y-values (such as T/HIS ERR function) does not measure correlation in the same way as the human eye. The T/HIS correlation function attempts to include and quantify the more subtle ways in which the correlation of two curves may be judged. The correlation can be calculated using either a strict or loose set of input parameters. The degree of correlation is rated between 0 and 100.

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a>	2nd <a href="#">Curve</a>
Correlation type	string	Correlation type, strict or loose

## Return type

Correlation value.

## Example

Calculate the correlation between curves m and n using the strict input parameters.

```
val = Operate.Cor(m,n,"strict");
```

---

**Cor3**(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)], X axis factor (optional)[*float*], Y axis factor (optional)[*float*]) [*static*]

## Description

Curve Correlation function. This function first normalises the curves using two factors either specified by the user or defaults calculated by the program (the maximum absolute X and Y values of both graphs). For each point on the first normalised curve, the shortest distance to the second normalised curve is calculated. The root mean square value of all these distances is subtracted from 1 and then multiplied by 100 to get an index between 0 and 100. The process is repeated along the second curve and the two indices are averaged to get a final index. The higher the index the closer the correlation between the two curves.

Note that the choice of normalising factors is important. Incorrect factors may lead to a correlation index outside the range of 0 to 100

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a>	2nd <a href="#">Curve</a>
X axis factor (optional)	float	Normalising factor used for X axis values
Y axis factor (optional)	float	Normalising factor used for Y axis values

## Return type

Correlation value.

## Example

Calculate the correlation between curves m and n using the default normalising factors.

```
val = Operate.Cor3(m,n);
```

Calculate the correlation between curves m and n using 0.1 and 1000.0 as the X and Y normalising factors.

```
val = Operate.Cor3(m,n,0.1,1000);
```

---

**Cos**(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [*static*]

## Description

Calculate Cosine

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate Cosine() of curve m and store as curve p  
`p = Operate.Cos(m) ;`

---

Da(Input Curve/[Curve](#)], Output Curve (optional)/[Curve](#)) [static]

## Description

Convert displacment spectrum to an acceleration spectrum

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Convert curve m and store as curve p  
`p = Operate.Da(m) ;`

---

Dif(Input Curve/[Curve](#)], Output Curve (optional)/[Curve](#)) [static]

## Description

Differentiate a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To differentiate curve m and store as curve p  
`p = Operate.Dif(m) ;`

---

Div(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or real], Output Curve (optional)[[Curve](#)] [static]

## Description

Divide Y axis values

### Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To divide the Y axis values for curve n by curve m and store as curve p  
`p = Operate.Div(m,n) ;`

To devide the Y axis values in curve m by 20.0 and store as curve p  
`p = Operate.Div(m,20.0) ;`

---

Dix(1st Curve[[Curve](#)], 2nd Curve or constant[[Curve](#) or real], Output Curve (optional)[[Curve](#)] [static]

## Description

Divide X axis values

### Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL



## Example

To divide the X axis values for curve n by curve m and store as curve p  
`p = Operate.Dix(m,n) ;`

To divide the X axis values in curve m by 20.0 and store as curve p  
`p = Operate.Dix(m,20.0) ;`

---

**Ds**(Input Curve[[Curve](#)], Broadening Factor[*float*], Redefine Frequencies[*string*], Output Curve (optional)[[Curve](#)] [static]

## Description

Generate a design spectrum from a response spectrum

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Broadening Factor	float	Spectrum broadening factor
Redefine Frequencies	string	T-HIS selects a new set of frequencies for the output (yes or no)
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Convert curve m and let T-HIS determine the new frequencies, store as curve p  
`p = Operate.Ds(m, "yes") ;`

---

**Dv**(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)] [static]

## Description

Convert displacement spectrum to a velocity spectrum

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Convert curve m and store as curve p  
`p = Operate.Dv(m) ;`

---

Env(Curve array[*array*], Output Curve (optional)[*Curve*] [static]

### Description

Generate an Envelope that bounds the min and max values of a group of curves

### Arguments

Name	Type	Description
Curve array	array	Array of <a href="#">Curve</a> objects
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Envelope of curves stored in curve array x and store as curve p  
`p = Operate.Env(x) ;`

---

Err(1st Curve[*Curve*], 2nd Curve[*Curve* or *real*], Output Curve (optional)[*Curve*] [static]

### Description

Calculate the degree of correlation between 2 curves

### Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a> or <i>real</i>	2nd <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

To calculate the correlation between curves n and m and store as curve p  
`p = Operate.Err(m,n) ;`

---

Exc(Input Curve[*Curve*], Output option[*string*], Output Curve (optional)[*Curve*] [static]

### Description

Calculate and displays an EXCeedence plot. This is a plot of force (Y axis) versus cumulative time (X axis) for which the force level has been exceeded. By default the Automatic option will create an exceedence plot using either the +ve OR the -ve values depending on which the input curve contains most of.

The Positive option will calculate the exceedence plot using only the points with +ve y values.

The Negative option will calculate the exceedence plot using only the points with -ve y values.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output option	string	Select between automatic, positive or negative.
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate Exceedence plot for curve m, using the positive option and store as curve p  
`p = Operate.Exc(m, "positive");`

---

**Exp**(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate E to the power of Y axis values

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate E to the power of Y axis values for curve m and store as curve p  
`p = Operate.Exp(m);`

---

**Fft**(Input Curve[[Curve](#)], Output option[*string*], X axis interval (optional)[*float*], Output option[*string*]) [static]

## Description

Fast Fourier Transform

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output option	string	Generate magnitude, magnitude+phase or real+imaginary, (one of magnitude,phase,real)
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output option	string	Scaling option, (either one or two)

## Return type

[Curve](#) object/array or NULL

## Example

Generate magnitude and phase curves and return a curve array. Regularise the input curve using an interval of 0.0001 first and scale using option two.

```
c_array = Operate.Fft(m, "phase", 0.0001, "one");
mag_curve = c_array[0];
phase_curve = c_array[1];
```

---

**Fir**(Input Curve[[Curve](#)], X axis interval (optional)[*float*], Output Curve (optional)[[Curve](#)]) [static]

## Description

FIR Filter

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Filter curve m and output as curve p. Regularise the input curve using an interval of 0.0001 first.

```
p = Operate.Fir(m, 0.0001);
```

---

**Hic**(Input Curve[[Curve](#)], Window[*float*], Acceleration factor[*float*]) [static]

## Description

HIC Calculation. After calculating the HIC value for a curve the value can also be obtained from the curve using the [Curve.hic](#) property. In addition to the HIC value the start and end time for the time window can also be obtained using the [Curve.hic\\_tmin](#) and [Curve.hic\\_tmax](#) properties.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Window	float	Maximum time window
Acceleration factor	float	Factor required to divide input acceleration curve by to convert to (G)

## Return type

HIC value.

### Example

Calculate HIC for curve m, using a window of 0.036s and a factor of 9810.

```
val = Operate.Hic(m,0.036,9810.0);
```

---

Hicd(Input Curve[[Curve](#)], Window[*float*], Acceleration factor[*float*]) [static]

## Description

Modified HIC(d) Calculation for free motion headform. After calculating the HIC value for a curve the value can also be obtained from the curve using the [Curve.hicd](#) property. In addition to the HIC(d) value the start and end time for the time window can also be obtained using the [Curve.hicd\\_tmin](#) and [Curve.hicd\\_tmax](#) properties.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Window	float	Maximum time window
Acceleration factor	float	Factor required to divide input acceleration curve by to convert to (G)

## Return type

HIC(d) value.

### Example

Calculate HIC(d) for curve m, using a window of 0.036s and a factor of 9810.

```
val = Operate.Hicd(m,0.036,9810.0);
```

---

Ifft(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)], Input type[*string*]) [static]

## Description

Inverse Fast Fourier Transform

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a>	2nd <a href="#">Curve</a>
Input type	string	Specifies if inputs are magnitude+phase or real+imaginary, (magnitude or real)

## Return type

[Curve](#) object or NULL

## Example

Generate curve from magnitude (m) and phase (p) data and return as curve q.  
`q = Operate.Ifft(m,p,"magnitude");`

---

Int(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Integrate a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To integrate curve m and store as curve p  
`p = Operate.Int(m);`

---

Log(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Natural Log of Y axis values

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate Natural Log of Y axis values for curve m and store as curve p  
`p = Operate.Log(m) ;`

---

Log10(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Log (base 10) of Y axis values

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate Log (base 10) of Y axis values for curve m and store as curve p  
`p = Operate.Log10(m) ;`

---

Log10x(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Log (base 10) of X axis values

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate Log (base 10) of X axis values for curve m and store as curve p  
`p = Operate.Log10x(m) ;`

---

Logx(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Natural Log of X axis values

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Calculate Natural Log of X axis values for curve m and store as curve p  
`p = Operate.Logx(m) ;`

---

**Lsq**(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

### Description

Calculate Least Squares Fit for a curve

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

To calculate Least Squares Fit for curve m and store as curve p  
`p = Operate.Lsq(m) ;`

---

**Map**(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or *real*], Output Curve (optional)[[Curve](#)]) [static]

### Description

Map Y axis values from one curve onto another curve

### Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a> or <i>real</i>	2nd <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL



## Example

To map curve n onto curve m and store as curve p  
`p = Operate.Map(m,n) ;`

---

## Max(Curve array[*array*], Output Curve (optional)[*Curve*]) [static]

### Description

Maximum of a group of curves

### Arguments

Name	Type	Description
Curve array	array	Array of <a href="#">Curve</a> objects
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Maximum of curves stored in curve array x  
`p = Operate.Max(x) ;`

---

## Min(Curve array[*array*], Output Curve (optional)[*Curve*]) [static]

### Description

Minimum of a group of curves

### Arguments

Name	Type	Description
Curve array	array	Array of <a href="#">Curve</a> objects
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Minimum of curves stored in curve array x  
`p = Operate.Min(x) ;`

---

## Mon(Input Curve[*Curve*], Output Curve (optional)[*Curve*]) [static]

### Description

Sort a curve into monotonically increasing X axis values.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To sort curve m and store as curve p  
`p = Operate.Mon(m) ;`

---

Mul(1st Curve[\[Curve\]](#), 2nd Curve or constant[\[Curve or real\]](#), Output Curve (optional)[\[Curve\]](#) [static]

## Description

Multiply Y axis values

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To multiply the Y axis values for curve n from m and store as curve p  
`p = Operate.Mul(m,n) ;`

To multiply the Y axis values in curve m by 20.0 and store as curve p  
`p = Operate.Mul(m,20.0) ;`

---

Mux(1st Curve[\[Curve\]](#), 2nd Curve or constant[\[Curve or real\]](#), Output Curve (optional)[\[Curve\]](#) [static]

## Description

Multiply X axis values

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To multiply the X axis values for curve n from m and store as curve p  
`p = Operate.Mux(m,n) ;`

To multiply the X axis values in curve m by 20.0 and store as curve p  
`p = Operate.Mux(m,20.0) ;`

---

## Ncp(1st Curve[[Curve](#)], 2nd Curve[[Curve](#)]) [static]

## Description

Calculate a plastic rotation curve for a beam from a moment/time and rotation/time

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	Moment / Time <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a>	Rotation /Time <a href="#">Curve</a>

## Return type

[Curve](#) object or NULL

## Example

Calculate plastic rotation curve p using curves m and r.  
`q = Operate.Ncp(m,r) ;`

---

## Nij(Shear Force[[Curve](#)], Axial Force[[Curve](#)], Moment[[Curve](#)], Fzc(tension)[*float*], Fzc(compression)[*float*], Myc(Flexion)[*float*], Myc(Extension)[*float*], E[*float*]) [static]

## Description

Biomechanical neck injury predictor. Used as a measure of injury due to the load transferred through the occipital condyles.

This function returns an array containing 4 curve objects.

Curve 1 - "Nte" is the tension-extension condition

Curve 2 - "Ntf" is the tension-flexion condition

Curve 3 - "Nce" is the compression-extension condition

Curve 4 - "Ncf" is the compression-flexion condition.

## Arguments

Name	Type	Description
Shear Force	<a href="#">Curve</a>	Shear Force <a href="#">Curve</a>
Axial Force	<a href="#">Curve</a>	Axial Force <a href="#">Curve</a>
Moment	<a href="#">Curve</a>	Moment <a href="#">Curve</a>
Fzc(tension)	float	Critical Axial Force (Tension)
Fzc(compression)	float	Critical Axial Force (Compression)
Myc(Flexion)	float	Critical bending moment (Flexion)
Myc(Extension)	float	Critical bending moment (Extension)
E	float	Distance

## Return type

Array of [Curve](#) objects. 1st curve : Nte curve 2nd curve : Ntf curve 3rd curve : Nce curve 4th curve : Ncf curve 2nd curve : PHD curve=

## Example

Calculate NIJ curves using input curves x,y,z, and constnats Fxc=1.0/2.0, Myc=3.0/4.0 and E=0.0.  
`c_array = Operate.Nij (x,y,z,1.0,2.0,3.0,4.0,0.0) ;`

---

Nor(Input Curve([Curve](#)), Output Curve (optional)([Curve](#))) [static]

## Description

Normalise Y axis values.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Normalise Y axis values of curve m and store as curve p  
`p = Operate.Nor (m) ;`

---

Nox(Input Curve([Curve](#)), Output Curve (optional)([Curve](#))) [static]

## Description

Normalise X axis values.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Normalise X axis values of curve m and store as curve p  
`p = Operate.NoX(m) ;`

---

Octave(Input Curve[\[Curve\]](#), Band type to convert to[\[String\]](#), Output Type[\[String\]](#), Output Type[\[String\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

### Description

Coverts a narrow band curve to either Octace or 1/3rd Octave bands

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Band type to convert to	String	Band type to convert to. Either "Octave" or "Third" Octave.
Output Type	String	Generate curve containing either "RMS" or "mean" values.
Output Type	String	Input curve contains either "Linear" or "dB" values.
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Convert curve m that contains Linear values to 1/3 Octave bands and output RMS in curve p  
`p = Operate.Ocatve(m, "third", "rms", "linear") ;`

---

Order(Input Curve[\[Curve\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

### Description

Reverse the order of points in a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Reverse the order of points in curve m and store as curve p  
`p = Operate.Order(m) ;`

---

**Pbut**(Input Curve[[Curve](#)], Frequency[*float*], Order[*integer*], X axis interval (optional)[*float*], Output Curve (optional)[[Curve](#)]) [static]

## Description

Pure Butterworth Filter

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Frequency	float	Cut-off Frequency (Hz)
Order	integer	Filter order
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Filter curve m using a cut-off of 400Hz and order 2 and output as curve p. Regularise the input curve using an interval of 0.0001 first.  
`p = Operate.Pbut(m,400.0,2,0.0001) ;`

---

**Power**(Input Curve[[Curve](#)], Power[*float*], Output Curve (optional)[[Curve](#)]) [static]

## Description

Raise to the power

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Power	float	Power to raise Y axis values by
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Raise the Y axis values for curve m to the power 2.5 and store as curve p  
`p = Operate.Power(m, 2.5) ;`

---

Rave(Input Curve/[Curve](#)), Output Curve (optional)/[Curve](#)) [static]

## Description

Calculate rolling average of a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate rolling average of curve m and store as curve p  
`p = Operate.Rave(m) ;`

---

Rec(Input Curve/[Curve](#)), Output Curve (optional)/[Curve](#)) [static]

## Description

Calculate reciprocal

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

### Example

Calculate reciprocal of curve m and store as curve p  
`p = Operate.Rec(m) ;`

---

**Reg**(Input Curve[\[Curve\]](#), X axis interval[\[float\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

## Description

Regularise X axis intervals for a curve.

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X axis interval	float	New X axis interval
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

### Example

Regularise curve m using a new X axis interval of 0.0001.  
`p = Operate.Reg(m, 0.0001) ;`

---

**Res**(Curve array[\[array\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

## Description

Resultant of a group of curves

### Arguments

Name	Type	Description
Curve array	array	Array of <a href="#">Curve</a> objects
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

### Example

Resultant of curves stored in curve array x  
`p = Operate.Res(x) ;`

---



---

Rev(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Reverse X and Y axis values

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Reverse X and Y axis values of curve m and store as curve p  
 p = Operate.Rev(m) ;

---

Rs(Input Curve[[Curve](#)], Damping Factor[[float](#)], Sampling Points[[int](#)], X axis interval (optional)[[float](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Generate a reponse spectrum from input accelerations

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Damping Factor	float	Dammping factor
Sampling Points	int	Number of points to sample over (30 or 70)
X axis interval (optional)	float	If defined then T-HIS will automatically regularise the curve using this value first
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Generate a response spectrum using a factor of 0.05 and 70 sampling points. Regularise the input curve using an interval of 0.0001 first.  
 p = Operate.Rs(m, 0.05, 70, 0.0001) ;

---

Sin(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Sine

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate Sine() of curve m and store as curve p  
`p = Operate.Sin(m) ;`

---

Smooth(Input Curve[\[Curve\]](#), Smoothing Factor[\[integer\]](#), Output Curve (optional)[\[Curve\]](#))  
 [static]

## Description

Apply a smoothing factor to a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Smoothing Factor	integer	Number of points to average over
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Smooth curve m using 7 points and store as curve p  
`p = Operate.Smooth(m, 7) ;`

---

Sqr(Input Curve[\[Curve\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

## Description

Square root of a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

---

## Return type

[Curve](#) object or NULL

### Example

Square root curve m and store as curve p  
`p = Operate.Sqr(m) ;`

---

Stress(Input Curve[\[Curve\]](#), Convert to[\[string\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

## Description

Convert between true and engineering stress

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Convert to	string	Type to convert to (True or Engineering)
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

### Example

Convert curve m from engineering to true stress and store as curve p  
`p = Operate.Stress(m, "True") ;`

---

Sub(1st Curve[\[Curve\]](#), 2nd Curve or constant[\[Curve or real\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

## Description

Subtract Y axis values

### Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To subtract the Y axis values for curve n from m and store as curve p  
`p = Operate.Sub(m,n) ;`

To subtract 20.0 from the Y axis values in curve m and store as curve p  
`p = Operate.Sub(m,20.0) ;`

---

**Sum(Curve array[*array*], Output Curve (optional)[*Curve*]) [static]**

## Description

Sum of a group of curves

## Arguments

Name	Type	Description
Curve array	array	Array of <a href="#">Curve</a> objects
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Sum of curves stored in curve array x  
`p = Operate.Res(x) ;`

---

**Sux(1st Curve[*Curve*], 2nd Curve or constant[*Curve* or *real*], Output Curve (optional)[*Curve*]) [static]**

## Description

Subtract X axis values

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve or constant	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a> or constant
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

To subtract the X axis values for curve n from m and store as curve p  
`p = Operate.Sux(m,n) ;`

To subtract 20.0 from the X axis values in curve m and store as curve p  
`p = Operate.Sux(m,20.0) ;`

---

Tan(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Calculate Tangent

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate Tangent() of curve m and store as curve p

```
p = Operate.Tan(m) ;
```

---

Thiv(X Acceleration[[Curve](#)], Y Acceleration[[Curve](#)], Yaw Rate[[Curve](#)], Dx[*float*], Dy[*float*], X0[*float*]) [static]

## Description

Theoretical Head Impact Velocity and the Post Impact Head Deceleration. These values are used to assess the performance of road side crash barriers.

This function returns an array containing 2 curve objects. The 1st curve is the THIV curve and the 2nd is the PHD curve. The peak values of these curves are the corresponding THIV and PHD values and can be obtained using the [Curve.ymax](#) property.

## Arguments

Name	Type	Description
X Acceleration	<a href="#">Curve</a>	X Acceleration <a href="#">Curve</a>
Y Acceleration	<a href="#">Curve</a>	Y Acceleration <a href="#">Curve</a>
Yaw Rate	<a href="#">Curve</a>	Yaw Rate <a href="#">Curve</a>
Dx	float	Horizontal distance between occupants head and vehicle
Dy	float	Lateral distance between occupants head and vehicle
X0	float	Horizontal distance between occupants head and vehicle CofG

## Return type

Array of [Curve](#) objects. 1st curve : THIV curve 2nd curve : PHD curve=

## Example

Calculate THIV and PHD curves x,y,z and distances Dx=0.6, Dy=0.3, X0=0.0.

```
c_array = Operate.Thiv(x,y,z,0.6,0.3,0.0) ;
```

```
thiv = c_array[0].ymax;
```

```
phd = c_array[1].ymax;
```

Tms(Input Curve[[Curve](#)], Period[*float*]) [static]

## Description

3ms Clip Calculation. After calculating the 3ms clip value for a curve the value can also be obtained from the curve using the [Curve.tms](#) property. In addition to the 3ms clip value the start and end time for the time window can also be obtained using the [Curve.tms\\_tmin](#) and [Curve.tms\\_tmax](#) properties.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Period	float	Clip period

## Return type

3ms Clip value.

## Example

Calculate 3ms clip for curve m, using a clip period of 0.003s.  
`val = Operate.Tms(m, 0.003);`

---

Translate(Input Curve[[Curve](#)], X value[*float*], Y value[*float*], Output Curve (optional)[[Curve](#)]) [static]

## Description

Translate a curve

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
X value	float	X translation value
Y value	float	Y translation value
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Translate curve m by x=0.2, y=0.3 and store as curve p  
`p = Operate.Translate(m, 0.2, 0.3);`

---

Tti(Upper Rib Acceleration[[Curve](#)], Lower Rib Acceleration[[Curve](#)], T12 Acceleration[[Curve](#)]) [static]

## Description

Thorax Trauma Index.

## Arguments

Name	Type	Description
Upper Rib Acceleration	<a href="#">Curve</a>	Upper Rib Acceleration <a href="#">Curve</a>
Lower Rib Acceleration	<a href="#">Curve</a>	Lower Rib Acceleration <a href="#">Curve</a>
T12 Acceleration	<a href="#">Curve</a>	T12 Acceleration <a href="#">Curve</a>

### Return type

TTI value.

### Example

Calculate TTI using curves x,y and z as inputs.

```
val = Operate.TTi(x,y,z);
```

---

Va(Input Curve[\[Curve\]](#), Output Curve (optional)[\[Curve\]](#), Input Curve[\[Curve\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

### Description

Convert acceleration spectrum to a velocity spectrum

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Convert curve m and store as curve p

```
p = Operate.Av(m);
```

---

Vc(Input Curve[\[Curve\]](#), A[\[float\]](#), B[\[float\]](#), Calculation method[\[string\]](#), Output Curve (optional)[\[Curve\]](#)) [static]

### Description

Viscous Criteria calculate. The VC calculation can be done using 2 different calculation methods ECER95 and IIHS.

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
A	float	Constant A
B	float	Constant B
Calculation method	string	Either ECER95 or IIHS.
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate VC for curve m, using A=1.3, B=0.229 and the ECER95 method  
`p = Operate.Vc(m,1.3,0.229,"ECER95");`

---

Vd(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Convert velocity spectrum to a displacement spectrum

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Convert curve m and store as curve p  
`p = Operate.Vd(m);`

---

Vec(1st Curve[[Curve](#)], 2nd Curve[[Curve](#) or *real*], 3rd Curve[[Curve](#) or *real*], Output Curve (optional)[[Curve](#)]) [static]

## Description

Vector magnitude of 3 curves



## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a>
3rd Curve	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate vector magnitude of curves m,n,o and store as curve p  
`p = Operate.Vec(m,n,o) ;`

---

`Vec2d(1st Curve[Curve], 2nd Curve[Curve or real], Output Curve (optional)[Curve])` [static]

## Description

Vector magnitude of 2 curves

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a> or real	2nd <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Calculate vector magnitude of curves m and n and store as curve p  
`p = Operate.Vec2d(m,n) ;`

---

`Wif(1st Curve[Curve], 2nd Curve[Curve])` [static]

## Description

Weighted Integrated Factor (WIFAC) Correlation function.

## Arguments

Name	Type	Description
1st Curve	<a href="#">Curve</a>	1st <a href="#">Curve</a>
2nd Curve	<a href="#">Curve</a>	2nd <a href="#">Curve</a>

## Return type

Correlation value.

### Example

Calculate the correlation between curves m and n.

```
val = Operate.Wif(m,n) ;
```

---

Window(Input Curve[[Curve](#)], Window Type[*string*], %age lead in (optional - cosine only)[*float*], Output Curve (optional)[[Curve](#)] [static]

## Description

Apply a smoothing window to a curve

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Window Type	string	Window type to apply (Hanning, cosine or exponential)
%age lead in (optional - cosine only)	float	%age lead in for cosine window
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

### Example

Apply a hanning window to curve m and store as curve p

```
p = Operate.Window(m, "Hanning") ;
```

---

Zero(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)] [static]

## Description

Translate curve to 0,0

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Translate curve m to (0,0) and store as curve p  
`p = Operate.Zero(m) ;`

---

ZeroX(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Translate curve to X=0.0

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Translate curve m to X=0 and store as curve p  
`p = Operate.ZeroX(m) ;`

---

ZeroY(Input Curve[[Curve](#)], Output Curve (optional)[[Curve](#)]) [static]

## Description

Translate curve to Y=0.0

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

## Return type

[Curve](#) object or NULL

## Example

Translate curve m to Y=0 and store as curve p  
`p = Operate.ZeroY(m) ;`

---

dB(Input Curve[[Curve](#)], Reference Value[*float*], Output Curve (optional)[[Curve](#)]) [static]

## Description

Converts a curve to dB ( $y = 20.0 * \log(y/yref)$ )

---

## Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Reference Value	float	Reference value
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Convert curve m to dB's using a reference value of 10.0 and store as curve p  
`p = Operate.dB(m, 10.0);`

---

`dBA(Input Curve[Curve], Weighting Type[String], Output Curve (optional)[Curve])` [static]

### Description

Applies A-weighting to a curve (converts from dB to dBA)

### Arguments

Name	Type	Description
Input Curve	<a href="#">Curve</a>	Input <a href="#">Curve</a>
Weighting Type	String	Apply either Narrow band (narrow) or Octave band (octave) A weighting
Output Curve (optional)	<a href="#">Curve</a>	<a href="#">Curve</a> to overwrite

### Return type

[Curve](#) object or NULL

### Example

Apply narrow band A-weighting to convert curve m from dB to dBA and store as curve p  
`p = Operate.dBA(m, "narrow");`

---

## PopupWindow class

The PopupWindow class allows you to create popup windows for a graphical user interface. [More...](#)

### Detailed Description

The PopupWindow class allows you to make popup windows (that you can place [Widgets](#) in) and link them to [Widgets](#). The popup window is then displayed by right clicking on the [Widget](#) the popup is linked to. The following very simple example shows how to create a popup window and link it to a label Widget.

```
// Create popup window
var pw = new PopupWindow();
// Create some widgets in the popup window
var pl = new Widget(pw, Widget.LABEL, 1, 30, 1, 7, "Label");
var pb = new Widget(pw, Widget.BUTTON, 1, 30, 7, 13, "Button");
var pt = new Widget(pw, Widget.TEXTBOX, 1, 30, 20, 26, "Textbox");
// Create window with title "Popup example" from 0.8-1.0 in x and 0.5-0.6 in y
var w = new Window("Popup example", 0.8, 1.0, 0.5, 0.6);
// Create label widget
var l = new Widget(w, Widget.LABEL, 1, 50, 1, 7, "Right click for popup...");
// link popup window to widget
l.popupWindow = pw;
// Assign the onPopup callback method to the function 'do_popup'
// This is only required if you want to make any changes before the popup
// appears
l.onPopup = do_popup;
// Show the widget and start event loop
w.Show();
////////////////////////////////////
function do_popup()
{
    Message("Showing popup");
}
```

See the documentation below and the [Widget](#) class for more details.

### Constructor

`new PopupWindow()`

### Description

Create a new [PopupWindow](#) object.

### Arguments

No arguments

### Return type

[PopupWindow](#) object

### Example

To create a PopupWindow containing the buttons "Create" and "Edit" and link it to button b:

```
var pw = new PopupWindow();
var c = new Widget(pw, Widget.BUTTON, 1, 30, 1, 7, "Create");
var e = new Widget(pw, Widget.BUTTON, 1, 30, 7, 13, "Edit");
b.popupWindow = pw;
```

## Symbol class

The Symbol class contains constants relating to curve symbols. [More...](#)

### Symbol constants

Name	Description
Symbol.CIRCLE	Circle symbol
Symbol.CROSS	Cross symbol
Symbol.DIAMOND	Diamond symbol
Symbol.DOT	Dot symbol
Symbol.HOURLASS	Hourglass symbol
Symbol.NONE	No symbol
Symbol.SQUARE	Square symbol
Symbol.STAR	Star symbol
Symbol.TRIANGLE	Triangle symbol

### Detailed Description

The Symbol class is used to define the symbol style used by curves:

```
p.symbol = Symbol.TRIANGLE;
```

## Units class

The Units class contains constants relating to curve units. [More...](#)

### Class functions

- [USER](#)(mass[*float*], time[*float*], length[*float*], angle[*float*], temperature[*float*])

### Units constants

Name	Description
Units.ACCELERATION	Acceleration units
Units.AREA	Area units
Units.DENSITY	Density units
Units.DISPLACEMENT	Displacement units
Units.ENERGY	Energy units
Units.ENERGY_DENSITY	Energy Density units
Units.FLUX	Thermal Flux units
Units.FORCE	Force units
Units.FORCE_WIDTH	Force per unit width units
Units.FREQUENCY	Frequency units
Units.LENGTH	Length units
Units.MASS	MAss units
Units.MASS_FLOW	Mass Flow rate units
Units.MOMENT	Moment units
Units.MOMENTUM	Momentum units
Units.MOMENT_WIDTH	Moment per unit width units
Units.NONE	No units
Units.POWER	Power units
Units.PRESSURE	Pressure units
Units.ROTATION	Rotation units
Units.ROTATIONAL_ACCELERATION	Rotational Acceleration units
Units.ROTATIONAL_VELOCITY	Rotational Velocity units
Units.STRAIN	Strain units
Units.STRESS	Stress units
Units.TEMPERATURE	Temperature units
Units.TIME	Time units
Units.UNKNOWN	Unknown units
Units.VELOCITY	Velocity units
Units.VOLUME	Volume units
Units.WORK	Work units

## Detailed Description

The Units class is used to define the units for each axis of a curve:

```
p.x_axis_units = Units.LENGTH
```

## Details of functions

USER(mass[*float*], time[*float*], length[*float*], angle[*float*], temperature[*float*]) [static]

### Description

Setup a user defined UNIT

### Arguments

Name	Type	Description
mass	float	Power for mass dimensions.
time	float	Power for time dimensions.
length	float	Power for length dimensions.
angle	float	Power for angle dimensions.
temperature	float	Power for temperature dimensions.

### Return type

0 (user defined)

### Example

To set the y-axis unit of curve l to (m/s)^2:

```
l.y_unit = Units.USER(0.0,2.0,-2.0,0.0,0.0);
```

---



---

## UnitSystem class

The UnitSystem class contains constants relating to curve unit systems. [More...](#)

### UnitSystem constants

Name	Description
UnitSystem.U1	U1 unit system (m,ks,s)
UnitSystem.U2	U2 unit system (mm,T,s)
UnitSystem.U3	U3 unit system (mm,kg,ms)
UnitSystem.U4	U4 unit system (mm,gm,ms)
UnitSystem.U5	U5 unit system (ft,slug,s)
UnitSystem.U6	U6 unit system (m,T,s)

### Detailed Description

The UnitSystem class is used to define the Unit System for a curve:

```
p.UnitSystem = UnitSystem.U1
```

## Widget class

The Widget class allows you to create components for a graphical user interface. [More...](#)

### Class functions

- [CtrlPressed\(\)](#)
- [PixelsPerUnit\(\)](#)
- [ShiftPressed\(\)](#)
- [StringLength](#)(text[*string*], monospace (optional)[*boolean*])

### Member functions

- [AddWidgetItem](#)(item[[WidgetItem](#)], position (optional)[*integer*])
- [Circle](#)(colour[*constant*], fill[*boolean*], xc[*integer*], yc[*integer*], radius[*integer*])
- [Clear\(\)](#)
- [Cross](#)(colour (optional)[*constant*])
- [Delete\(\)](#)
- [DumpImageString](#)(filename[*string*])
- [Hide\(\)](#)
- [ItemAt](#)(index[*integer*])
- [Line](#)(colour[*constant*], x1[*integer*], y1[*integer*], x2[*integer*], y2[*integer*])
- [Polygon](#)(colour[*constant*], fill[*boolean*], x1[*integer*], y1[*integer*], x2[*integer*], y2[*integer*], ... xn[*integer*], ... yn[*integer*])
- [ReadImageFile](#)(filename[*string*], justify (optional)[*constant*])
- [ReadImageString](#)(string[*string*], justify (optional)[*constant*])
- [Rectangle](#)(colour[*constant*], fill[*boolean*], x1[*integer*], y1[*integer*], x2[*integer*], y2[*integer*])
- [RemoveAllWidgetItems\(\)](#)
- [RemoveWidgetItem](#)(item[[WidgetItem](#)])
- [Show\(\)](#)
- [Static\(\)](#)
- [Tick](#)(colour (optional)[*constant*])
- [TotalItems\(\)](#)
- [WidgetItems\(\)](#)

### Widget constants

Name	Description
Widget.BUTTON	Button widget
Widget.CHECKBOX	Checkbox widget
Widget.COMBOBOX	Combobox widget
Widget.LABEL	Label widget
Widget.LISTBOX	Listbox widget
Widget.TEXTBOX	Text input widget

#### Constants for Justification

Name	Description
Widget.BOTTOM	Bottom justification
Widget.CENTRE	Centre (horizontal) justification
Widget.LEFT	Left justification
Widget.MIDDLE	Middle (vertical) justification
Widget.RIGHT	Right justification
Widget.TOP	Top justification

#### Constants for Selection

Name	Description
Widget.SELECT_ENHANCED	Multiple <a href="#">WidgetItems</a> in a <a href="#">ListBox</a> Widget can be selected. When the user selects a <a href="#">WidgetItem</a> the selection is cleared and the new <a href="#">WidgetItem</a> selected. However, if the user presses the Ctrl key when clicking on a <a href="#">WidgetItem</a> , the clicked <a href="#">WidgetItem</a> gets toggled and all other <a href="#">WidgetItems</a> are left untouched. If the user presses the Shift key while clicking on a <a href="#">WidgetItem</a> , all <a href="#">WidgetItems</a> between the last selected <a href="#">WidgetItem</a> and the clicked <a href="#">WidgetItem</a> are selected or unselected, depending on the state of the clicked <a href="#">WidgetItem</a> .
Widget.SELECT_MULTIPLE	Multiple <a href="#">WidgetItems</a> in a <a href="#">ListBox</a> Widget can be selected. When the user selects a <a href="#">WidgetItem</a> , the selection status of that <a href="#">WidgetItem</a> is toggled and the other <a href="#">WidgetItems</a> are left alone.
Widget.SELECT_NONE	No <a href="#">WidgetItem</a> in a <a href="#">ListBox</a> Widget can be selected
Widget.SELECT_SINGLE	A single <a href="#">WidgetItem</a> in a <a href="#">ListBox</a> Widget can be selected. When the user selects a <a href="#">WidgetItem</a> , any already-selected <a href="#">WidgetItem</a> becomes unselected, and the user cannot unselect the selected <a href="#">WidgetItem</a> by clicking on it.

## Constants for Colour

Name	Description
Widget.BLACK	Colour black
Widget.BLUE	Colour blue
Widget.CYAN	Colour cyan
Widget.DARKBLUE	Colour dark blue
Widget.DARKGREEN	Colour dark green
Widget.DARKGREY	Colour dark grey
Widget.DARKRED	Colour dark red
Widget.DEFAULT	Default colour for widgets
Widget.GREEN	Colour green
Widget.GREY	Colour grey
Widget.LIGHTGREY	Colour light grey
Widget.MAGENTA	Colour magenta
Widget.ORANGE	Colour orange
Widget.RED	Colour red
Widget.WHITE	Colour white
Widget.YELLOW	Colour yellow

## Widget properties

Name	Type	Description
active	logical	If widget is active (true) or disabled (false)
background	constant	Widget background colour. Can be: <a href="#">Widget.BLACK</a> , <a href="#">Widget.WHITE</a> , <a href="#">Widget.RED</a> , <a href="#">Widget.GREEN</a> , <a href="#">Widget.BLUE</a> , <a href="#">Widget.CYAN</a> , <a href="#">Widget.MAGENTA</a> , <a href="#">Widget.YELLOW</a> , <a href="#">Widget.DARKRED</a> , <a href="#">Widget.DARKGREEN</a> , <a href="#">Widget.DARKBLUE</a> , <a href="#">Widget.GREY</a> , <a href="#">Widget.DARKGREY</a> , <a href="#">Widget.LIGHTGREY</a> , <a href="#">Widget.ORANGE</a> , <a href="#">Widget.DEFAULT</a>
bottom	integer	Widget bottom coordinate

foreground	constant	Widget foreground colour. Can be: <a href="#">Widget.BLACK</a> , <a href="#">Widget.WHITE</a> , <a href="#">Widget.RED</a> , <a href="#">Widget.GREEN</a> , <a href="#">Widget.BLUE</a> , <a href="#">Widget.CYAN</a> , <a href="#">Widget.MAGENTA</a> , <a href="#">Widget.YELLOW</a> , <a href="#">Widget.DARKRED</a> , <a href="#">Widget.DARKGREEN</a> , <a href="#">Widget.DARKBLUE</a> , <a href="#">Widget.GREY</a> , <a href="#">Widget.DARKGREY</a> , <a href="#">Widget.LIGHTGREY</a> , <a href="#">Widget.ORANGE</a> , <a href="#">Widget.DEFAULT</a>
hover	string	Widget hover text
imageHeight (read only)	integer	Height of widget image (pixels)
imageWidth (read only)	integer	Width of widget image (pixels)
justify	constant	Widget justification. Can be: <a href="#">Widget.LEFT</a> , <a href="#">Widget.RIGHT</a> or <a href="#">Widget.CENTRE</a> (default).
left	integer	Widget left coordinate
lineWidth	integer	Width of lines when drawing graphics (initially 1; values 1-255 allowed).
macroTag	string	Tag to use for this widget when recording a macro. If empty then the <a href="#">text</a> property value will be used.
monospace	boolean	true if the widget uses a monospace font instead of a proportional width font (default). <a href="#">Label</a> and <a href="#">button</a> Widgets only.
onChange	function	Function to call when the text in a <a href="#">TEXTBOX</a> widget or the selection in a <a href="#">COMBOBOX</a> widget is changed. The Widget object is accessible in the function using the 'this' keyword (see the example below for more details of how to define the function and how to use the 'this' keyword). To unset the function set the property to null. <b>Note that this function is called when the user actually types something into the textbox, or selects an item in the combobox, NOT when the <a href="#">Widget.text</a> property changes.</b>
onClick	function	Function to call when a <a href="#">BUTTON</a> , <a href="#">CHECKBOX</a> or <a href="#">COMBOBOX</a> widget is clicked. The Widget object is accessible in the function using the 'this' keyword (see the example below for more details of how to define the function and how to use the 'this' keyword). To unset the function set the property to null. <b>Note that this function is called when the user actually clicks on the button, NOT when the <a href="#">Widget.pushed</a> property changes.</b> For the <a href="#">COMBOBOX</a> widget the function is called <b>before</b> the list of items is mapped.
onPopup	function	Function to call when a <a href="#">BUTTON</a> , <a href="#">LABEL</a> or <a href="#">TEXTBOX</a> widget is right clicked to map a popup. The <a href="#">Widget</a> object is accessible in the function using the 'this' keyword. The <a href="#">PopupWindow</a> can then be found by using the <a href="#">popupWindow</a> property of the <a href="#">Widget</a> . The function is called <b>before</b> the popup is mapped so you can change the widgets in the popup as required.
onTimer	function	Function to call for a widget when <a href="#">timerDelay</a> ms have elapsed after setting this. Additionally if <a href="#">timerRepeat</a> is set this function will be called repetitively, every <a href="#">timerDelay</a> ms. The Widget object is accessible in the function using the 'this' keyword. To unset the function set the property to null. <b>Note that as soon as this property is set the timer starts!</b>
popupDirection	constant	How <a href="#">PopupWindow</a> will be mapped relative to this widget. Can be <a href="#">Widget.LEFT</a> , <a href="#">Widget.RIGHT</a> , <a href="#">Widget.TOP</a> or <a href="#">Widget.BOTTOM</a> (default).
popupSymbol	logical	TRUE (default) if a symbol will be shown for a <a href="#">PopupWindow</a> .
popupWindow	<a href="#">PopupWindow</a> object	<a href="#">PopupWindow</a> for this Widget. Only available for <a href="#">Button</a> , <a href="#">Label</a> and <a href="#">Textbox</a> Widgets. To remove a <a href="#">PopupWindow</a> from a <a href="#">Widget</a> set to null.
pushed	logical	If widget is pushed (true) or not (false). This only affects <a href="#">Widget.BUTTON</a> with the <a href="#">Widget.toggle</a> property set, and <a href="#">Widget.CHECKBOX</a> widgets.
right	integer	Widget right coordinate

select	constant	Selection method for <a href="#">ListBox</a> Widgets. Can be: <a href="#">Widget.SELECT_NONE</a> , <a href="#">Widget.SELECT_SINGLE</a> or <a href="#">Widget.SELECT_MULTIPLE</a> or <a href="#">Widget.SELECT_ENHANCED</a> (default).
selectedItem	<a href="#">WidgetItem</a> object	<a href="#">WidgetItem</a> that is currently selected for a <a href="#">ComboBox</a> Widget. If null no <a href="#">WidgetItem</a> is selected. For a <a href="#">ListBox</a> Widget this property contains the last <a href="#">WidgetItem</a> that was (de)selected. To get a list of all of the selected <a href="#">WidgetItems</a> use <a href="#">WidgetItems()</a> to return all of the <a href="#">WidgetItems</a> and inspect the <a href="#">WidgetItem</a> <a href="#">selected</a> property.
shown (read only)	boolean	true if the widget is visible. To alter the visibility of a widget use the <a href="#">Show()</a> and <a href="#">Hide()</a> methods.
text	string	Widget text. For a <a href="#">ComboBox</a> Widget this will be the text for the currently selected <a href="#">WidgetItem</a>
textHidden	boolean	true if the widget text is hidden and replaced by asterisks. This may be used to create textboxes to type passwords in. <a href="#">TextBox</a> Widgets only.
timerDelay	integer	Delay in ms before the function set for <a href="#">onTimer</a> will be called. The initial value is 1000 (ms). Also see <a href="#">timerRepeat</a> .
timerRepeat	logical	If the function set for <a href="#">onTimer</a> will be called once (false) or repeatedly (true). The initial value is false. Also see <a href="#">timerDelay</a> .
toggle	logical	If widget can be toggled (true) or not (false). This only affects <a href="#">Widget.BUTTON</a> widgets.
top	integer	Widget top coordinate
window (read only)	<a href="#">Window</a> object	The <a href="#">Window</a> that this widget is defined in
xResolution	integer	X resolution of button when drawing <a href="#">lines</a> , <a href="#">circles</a> , <a href="#">polygons</a> and <a href="#">rectangles</a> (initially 100). X coordinates on the Widget can be from 0 (on the left of the widget) to xResolution (on the right of the widget). Available for <a href="#">Widget.LABEL</a> and <a href="#">Widget.BUTTON</a> Widgets.
yResolution	integer	Y resolution of button when drawing <a href="#">lines</a> , <a href="#">circles</a> , <a href="#">polygons</a> and <a href="#">rectangles</a> (initially 100). Y coordinates on the Widget can be from 0 (on the top of the widget) to yResolution (on the bottom of the widget). Available for <a href="#">Widget.LABEL</a> and <a href="#">Widget.BUTTON</a> Widgets.

## Detailed Description

The Widget class allows you to create Widgets (buttons, textboxes etc) in a [Window](#) for a graphical user interface. Callback functions can be declared for widgets to give actions when a button is pressed or the text in a textbox is selected etc. The following example displays various widgets in a window. Several callback methods are used. The exit button allows the user to exit the script but the button is only made active if the checkbox widget is ticked. If the button widgets are pressed feedback is given to the user

```
var count = 0;
// Create window
var w = new Window("Test", 0.8, 1.0, 0.5, 0.6);
// Create all of the widgets
var l = new Widget(w, Widget.LABEL, 1, 30, 1, 7, "Text:");
var t = new Widget(w, Widget.TEXTBOX, 31, 80, 1, 7, "Enter text");
var b = new Widget(w, Widget.BUTTON, 1, 30, 8, 14, "Press me");
var b2= new Widget(w, Widget.BUTTON, 31, 61, 8, 14, "Don't press me");
var c = new Widget(w, Widget.CHECKBOX, 62, 68, 8, 14);
var l2= new Widget(w, Widget.LABEL, 1, 80, 15, 21, "You haven't pressed the
button yet...");
var e = new Widget(w, Widget.BUTTON, 1, 21, 22, 28, "Exit");
// Allow button widget b2 to toggle
b2.toggle = true;
// The exit button is initially inactive
e.active = false;
// Assign the callback functions
b.onClick = clicked;
b2.onClick = clicked;
c.onClick = clicked;
t.onChange = changed;
e.onClick = confirm_exit;
// Show the window and start event loop
w.Show();
////////////////////////////////////
function clicked()
{
// If checkbox is clicked then set the state of the exit button
if (this === c)
{
    Message("Checkbox clicked");
    e.active = c.pushed;
}
// If the "Don't press me' button is pressed then change the colour if the
button is pressed in.
else if (this === b2)
{
    Message("I said don't press!!!");
    if (b2.pushed) b2.background = Widget.WHITE;
    else b2.background = Widget.DEFAULT;
}
// If the "Press me" button is pressed then update the text in the label widget
// with how many times the button has been pressed.
else
{
    Message("You pressed...");
    count++;
    l2.text = "Button pressed " + count + " times";
}
}
////////////////////////////////////
function changed()
{
// If the user has changed the text in the textbox then give a message in
// the dialogue box
    Message("Text has changed to " + this.text);
}
////////////////////////////////////
function confirm_exit()
{
// Map confirm box
    var ret = Window.Question("Confirm exit", "Are you sure you want to quit?");
// If the user has answered yes then exit from the script.
    if (ret == Window.YES) Exit();
}
```

## Constructor

new Widget(window[[Window](#) or [PopupWindow](#)], type[constant], left[integer], right[integer], top[integer], bottom[integer], text (optional for LABEL, BUTTON and TEXTBOX, not required for CHECKBOX, COMBOBOX and LISTBOX)[string])

## Description

Create a new [Widget](#) object.

## Arguments

Name	Type	Description
window	<a href="#">Window</a> or <a href="#">PopupWindow</a>	<a href="#">Window</a> or <a href="#">PopupWindow</a> that widget will be created in
type	constant	Widget type. Can be <a href="#">Widget.LABEL</a> , <a href="#">Widget.BUTTON</a> , <a href="#">Widget.CHECKBOX</a> , <a href="#">Widget.COMBOBOX</a> , <a href="#">Widget.LISTBOX</a> or <a href="#">Widget.TEXTBOX</a> .
left	integer	left coordinate of widget
right	integer	right coordinate of widget
top	integer	top coordinate of widget
bottom	integer	bottom coordinate of widget
text (optional for LABEL, BUTTON and TEXTBOX, not required for CHECKBOX, COMBOBOX and LISTBOX)	string	Text to show on widget

## Return type

[Widget](#) object

## Details of functions

AddWidgetItem(item[[WidgetItem](#)], position (optional)[integer])

## Description

Adds a [WidgetItem](#) to the [Widget](#). Also see [Widget.RemoveAllWidgetItems](#) and [Widget.RemoveWidgetItem](#).

## Arguments

Name	Type	Description
item	<a href="#">WidgetItem</a>	<a href="#">WidgetItem</a> to add
position (optional)	integer	Position on <a href="#">Widget</a> to add the <a href="#">WidgetItem</a> . Any existing <a href="#">WidgetItems</a> will be shifted down as required. If omitted the <a href="#">WidgetItem</a> will be added to the end of the existing ones. <b>Note that positions start at 0.</b>

## Return type

No return value

---

## Example

To add WidgetItem wi to widget w:  
`w.AddItem(wi);`

---

**Circle(colour[constant], fill[boolean], xc[integer], yc[integer], radius[integer])**

## Description

Draws a circle on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets. The coordinates are local to the Widget, not the Window. See properties [xResolution](#) and [yResolution](#) for more details. Note that the widget graphics will only be updated when the widget is redrawn. This is to allow the user to do multiple drawing commands on a widget. To force the widget to be redrawn call [Show\(\)](#).

## Arguments

Name	Type	Description
colour	constant	Colour of circle. See <a href="#">foreground</a> for colours.
fill	boolean	If circle should be filled or not.
xc	integer	x coordinate of centre of circle.
yc	integer	y coordinate of centre of circle.
radius	integer	radius of circle.

## Return type

no return value

## Example

To draw a red filled circle, radius 25, at (50, 50) on widget w:  
`w.Circle(Widget.RED, true, 50, 50, 25);`

---

## Clear()

## Description

Clears any graphics on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets. Note that the widget graphics will only be updated when the widget is redrawn. This is to allow the user to do multiple drawing commands on a widget. To force the widget to be redrawn call [Show\(\)](#).

## Arguments

No arguments

## Return type

no return value

## Example

To clear any graphics for widget w:  
`w.Clear();`



---

**Cross(colour (optional)[constant])****Description**

Draws a cross symbol on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets.

**Arguments**

Name	Type	Description
colour (optional)	constant	Colour of cross symbol. See <a href="#">foreground</a> for colours. If omitted, current foreground colour is used.

**Return type**

no return value

**Example**

To draw a red cross symbol on widget w:  
`w.Cross (Widget.RED) ;`

---

**CtrlPressed() [static]****Description**

Check to see if the Ctrl key is pressed

**Arguments**

No arguments

**Return type**

true/false

**Example**

To test if someone has the Ctrl key pressed:  
`if (Widget.CtrlPressed()) { ... }`

---

**Delete()****Description**

Deletes the widget from T-HIS (removing it from the window it is defined in) and returns any memory/resources used for the widget. This function should not normally need to be called. However, sometimes a script may want to recreate widgets in a window many times and unless the old widgets are deleted T-HIS will reach the maximum number of widgets for a window ([Options.max\\_widgets](#)). To avoid this problem this method can be used to force T-HIS to delete and return the resources for a widget. **Do not use the Widget object after calling this method.**

**Arguments**

No arguments

**Return type**

no return value

---

## Example

To delete widget *w*:  
`w.Delete()` ;

---

## DumpImageString(filename[*string*])

### Description

Dumps a string representation of an image for a widget to a file in a form that can be used by [Widget.ReadImageString\(\)](#). Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets.

### Arguments

Name	Type	Description
filename	string	Filename to dump string representation to

### Return type

no return value

### Example

To dump the image data to file 'image\_data' for widget *w*:  
`w.DumpImageString('image_data')` ;

---

## Hide()

### Description

Hides the widget on the screen

### Arguments

No arguments

### Return type

No return value

### Example

To hide widget *w*:  
`w.Hide()` ;

---

## ItemAt(index[*integer*])

### Description

Returns the [WidgetItem](#) object used at *index* in this Widget. See also [Widget.TotalItems\(\)](#) and [Widget.WidgetItems\(\)](#).

---

## Arguments

Name	Type	Description
index	integer	index to return <a href="#">WidgetItem</a> for. <b>Note that indices start at 0.</b>

## Return type

[WidgetItem](#) object.

## Example

```
To loop over the WidgetItems used in Widget w
for (i=0; i<w.TotalItems(); i++)
{
    wi = w.ItemAt(i);
}
```

---

## Line(colour[constant], x1[integer], y1[integer], x2[integer], y2[integer])

## Description

Draws a line on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets. The coordinates are local to the Widget, not the Window. See properties [xResolution](#) and [yResolution](#) for more details. Note that the widget graphics will only be updated when the widget is redrawn. This is to allow the user to do multiple drawing commands on a widget. To force the widget to be redrawn call [Show\(\)](#).

## Arguments

Name	Type	Description
colour	constant	Colour of line. See <a href="#">foreground</a> for colours.
x1	integer	x coordinate of start of line.
y1	integer	y coordinate of start of line.
x2	integer	x coordinate of end of line.
y2	integer	y coordinate of end of line.

## Return type

no return value

## Example

```
To draw a red line from (10, 90) to (90, 10) on widget w:
w.Line(Widget.RED, 10, 90, 90, 10);
```

---

## PixelsPerUnit() [static]

## Description

Returns the number of pixels per unit coordinate. This will vary depending on the monitor T-HIS is running on.

## Arguments

No arguments

## Return type

pixels/unit (float)

### Example

To return how many pixels there are per unit coordinate:

```
var ppu = Widget.PixelsPerUnit();
```

---

**Polygon**(colour[constant], fill[boolean], x1[integer], y1[integer], x2[integer], y2[integer], ... xn[integer], ... yn[integer])

## Description

Draws a polygon on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets. The coordinates are local to the Widget, not the Window. See properties [xResolution](#) and [yResolution](#) for more details. Note that the widget graphics will only be updated when the widget is redrawn. This is to allow the user to do multiple drawing commands on a widget. To force the widget to be redrawn call [Show\(\)](#).

## Arguments

Name	Type	Description
colour	constant	Colour of polygon. See <a href="#">foreground</a> for colours.
fill	boolean	If polygon should be filled or not.
x1	integer	x coordinate of point 1.
y1	integer	y coordinate of point 1.
x2	integer	x coordinate of point 2.
y2	integer	y coordinate of point 2.
... xn	integer	x coordinate of point n.
... yn	integer	y coordinate of point n.

Alternatively instead of x1, y1 etc you can specify a single argument which is an array of coordinates to use. In either case the number of points (x, y pairs) is limited to 30. Any extra points will be ignored.

## Return type

no return value

### Example

To draw a red filled triangle with corners (20, 20) and (50, 80) and (80, 20) on widget w:

```
w.Polygon(Widget.RED, true, 20, 20, 50, 80, 80, 20);
```

---

**ReadImageFile**(filename[string], justify (optional)[constant])

## Description

Reads an image from a file to show on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets. The image will be shown on the widget underneath any text. Note that due to the way that colours are used for menus in T-HIS only a small number of colours are available for Widget images. Black and white images will display without any issues but colour images will be displayed with a reduced set of colours.

## Arguments

Name	Type	Description
filename	string	Image file (BMP, GIF, JPEG or PNG) to read. To remove an image use null.
justify (optional)	constant	Widget justification. Can be a bitwise or of <a href="#">Widget.LEFT</a> , <a href="#">Widget.RIGHT</a> or <a href="#">Widget.CENTRE</a> and <a href="#">Widget.TOP</a> , <a href="#">Widget.MIDDLE</a> or <a href="#">Widget.BOTTOM</a> . If omitted the default is <a href="#">Widget.CENTRE Widget.MIDDLE</a> .

## Return type

no return value

## Example

To read image example.png for widget w and place it at the top left:

```
w.ReadImageFile("example.png", Widget.TOP|Widget.LEFT);
```

To remove an image from widget w:

```
w.ReadImageFile(null);
```

---

## ReadImageString(string[*string*], justify (optional)[*constant*])

## Description

Reads an image from a JavaScript string previously created by [Widget.DumpImageString\(\)](#) to show on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets. The image will be shown on the widget underneath any text. Note that due to the way that colours are used for menus in T-HIS only a small number of colours are available for Widget images. Black and white images will display without any issues but colour images will be displayed with a reduced set of colours.

## Arguments

Name	Type	Description
string	string	String containing the image data previously created by <a href="#">Widget.DumpImageString()</a> . To remove an image use null.
justify (optional)	constant	Widget justification. Can be a bitwise or of <a href="#">Widget.LEFT</a> , <a href="#">Widget.RIGHT</a> or <a href="#">Widget.CENTRE</a> and <a href="#">Widget.TOP</a> , <a href="#">Widget.MIDDLE</a> or <a href="#">Widget.BOTTOM</a> . If omitted the default is <a href="#">Widget.CENTRE Widget.MIDDLE</a> .

## Return type

no return value

## Example

To read image data from string s for widget w and place it at the top left:

```
w.ReadImageString(s, Widget.TOP|Widget.LEFT);
```

To remove an image from widget w:

```
w.ReadImageString(null);
```

---

---

**Rectangle**(colour[*constant*], fill[*boolean*], x1[*integer*], y1[*integer*], x2[*integer*], y2[*integer*])

## Description

Draws a rectangle on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets. The coordinates are local to the Widget, not the Window. See properties [xResolution](#) and [yResolution](#) for more details. Note that the widget graphics will only be updated when the widget is redrawn. This is to allow the user to do multiple drawing commands on a widget. To force the widget to be redrawn call [Show\(\)](#).

## Arguments

Name	Type	Description
colour	constant	Colour of rectangle. See <a href="#">foreground</a> for colours.
fill	boolean	If rectangle should be filled or not.
x1	integer	x coordinate of first corner of rectangle.
y1	integer	y coordinate of first corner of rectangle.
x2	integer	x coordinate of second (opposite) corner of rectangle.
y2	integer	y coordinate of second (opposite) corner of rectangle.

## Return type

no return value

## Example

To draw a red filled rectangle with corners (20, 20) and (80, 80) on widget w:  
`w.Rectangle(Widget.RED, true, 20, 20, 80, 80);`

---

## RemoveAllWidgetItems()

## Description

Removes any [WidgetItems](#) from the [Widget](#). Also see [Widget.AddItem](#) and [Widget.RemoveWidgetItem](#).

## Arguments

No arguments

## Return type

No return value

## Example

To remove all WidgetItems from widget w:  
`w.RemoveAllWidgetItems();`

---

## RemoveWidgetItem(item[[WidgetItem](#)])

## Description

Removes a [WidgetItem](#) from the [Widget](#). Also see [Widget.AddItem](#) and [Widget.RemoveAllWidgetItems](#).

## Arguments

Name	Type	Description
item	<a href="#">WidgetItem</a>	<a href="#">WidgetItem</a> to remove

## Return type

No return value

## Example

To remove WidgetItem wi from widget w:  
`w.RemoveWidgetItem(wi);`

---

## ShiftPressed() [static]

## Description

Check to see if the Shift key is pressed

## Arguments

No arguments

## Return type

true/false

## Example

To test if someone has the Shift key pressed:  
`if (Widget.ShiftPressed()) { ... }`

---

## Show()

## Description

Shows the widget on the screen

## Arguments

No arguments

## Return type

No return value

## Example

To show widget w:  
`w.Show();`

---

## Static()

### Description

[Windows](#) have two different regions for [Widgets](#). A 'normal' region which can be scrolled if required (if the window is made smaller scrollbars will be shown which can be used to scroll the contents) and a 'static' region at the top of the [Window](#) which is fixed and does not scroll. For an example of a static region in a [Window](#) see any of the keyword editing panels. The 'Dismiss', 'Create', 'Reset' etc buttons are in the static region. By default [Widgets](#) are put into the normal region of the [Window](#). This method puts the [Widget](#) to the static region of the [Window](#).

### Arguments

No arguments

### Return type

No return value

### Example

To put widget w in the static part of the window:  
`w.Static();`

---

## StringLength(text[*string*], monospace (optional)[*boolean*]) [static]

### Description

Returns the length of a string in Widget units. This can be used to find what size a Widget must be to be able to display the string.

### Arguments

Name	Type	Description
text	string	Text to find the width of
monospace (optional)	boolean	If true then width will be calculated using a monospace font. If false (default) then the normal proportional width font will be used

### Return type

integer

### Example

To get the width of string 'Example':  
`var len = Widget.StringLength('Example');`

---

## Tick(colour (optional)[*constant*])

### Description

Draws a tick symbol on the widget. Only possible for [Widget.LABEL](#) and [Widget.BUTTON](#) widgets.



---

## Arguments

Name	Type	Description
colour (optional)	constant	Colour of tick symbol. See <a href="#">foreground</a> for colours. If omitted, current foreground colour is used.

## Return type

no return value

## Example

To draw a red tick symbol on widget *w*:  
`w.Tick(Widget.RED);`

---

## TotalItems()

## Description

Returns the number of the [WidgetItem](#) objects used in this Widget (or 0 if none used). See also [Widget.ItemAt\(\)](#) and [Widget.WidgetItems\(\)](#).

## Arguments

No arguments

## Return type

integer

## Example

To return the total number of WidgetItems used for Widget *w*  
`var total = w.TotalItems();`

---

## WidgetItems()

## Description

Returns an array of the [WidgetItem](#) objects used in this Widget (or null if none used). See also [Widget.ItemAt\(\)](#) and [Widget.TotalItems\(\)](#).

## Arguments

No arguments

## Return type

array

## Example

To return WidgetItems used for Widget *w*  
`var wi = w.WidgetItems();`

---

## WidgetItem class

The WidgetItem class allows you to create items for combobox [Widgets](#). [More...](#)

### WidgetItem properties

Name	Type	Description
background	constant	Widget background colour. Can be: <a href="#">Widget.BLACK</a> , <a href="#">Widget.WHITE</a> , <a href="#">Widget.RED</a> , <a href="#">Widget.GREEN</a> , <a href="#">Widget.BLUE</a> , <a href="#">Widget.CYAN</a> , <a href="#">Widget.MAGENTA</a> , <a href="#">Widget.YELLOW</a> , <a href="#">Widget.DARKRED</a> , <a href="#">Widget.DARKGREEN</a> , <a href="#">Widget.DARKBLUE</a> , <a href="#">Widget.GREY</a> , <a href="#">Widget.DARKGREY</a> , <a href="#">Widget.LIGHTGREY</a> or <a href="#">Widget.DEFAULT</a>
foreground	constant	Widget foreground colour. Can be: <a href="#">Widget.BLACK</a> , <a href="#">Widget.WHITE</a> , <a href="#">Widget.RED</a> , <a href="#">Widget.GREEN</a> , <a href="#">Widget.BLUE</a> , <a href="#">Widget.CYAN</a> , <a href="#">Widget.MAGENTA</a> , <a href="#">Widget.YELLOW</a> , <a href="#">Widget.DARKRED</a> , <a href="#">Widget.DARKGREEN</a> , <a href="#">Widget.DARKBLUE</a> , <a href="#">Widget.GREY</a> , <a href="#">Widget.DARKGREY</a> , <a href="#">Widget.LIGHTGREY</a> or <a href="#">Widget.DEFAULT</a>
index (read only)	index	The index of this widgetitem in the parent widget (undefined if widgetitem is not assigned to a widget).
onClick	function	Function to call when a widget item in a <a href="#">COMBOBOX</a> or <a href="#">LISTBOX</a> widget is clicked. The Widgetitem object is accessible in the function using the 'this' keyword.
onMouseOver	function	Function to call when the mouse moves over a widget item in a <a href="#">COMBOBOX</a> or <a href="#">LISTBOX</a> widget. The Widgetitem object is accessible in the function using the 'this' keyword.
selectable	logical	If the widget item can be selected (true) or not (false).
selected (read only)	logical	If the widget item is selected (true) or not (false).
text	string	Widget text
widget (read only)	object	The widget that this item is defined for

## Detailed Description

The `WidgetItem` class allows you to create items for combobox Widgets in a [Window](#) for a graphical user interface. The following example shows how `WidgetItems` are used to create a Combobox Widget and how to assign callbacks to determine when the selection has been changed.

```
var items = ["D3PLOT", "PRIMER", "SHELL", "REPORTER", "T/HIS"]
// Create window
var w = new Window("Combobox example", 0.8, 1.0, 0.5, 0.6);
// A simple combobox with a few items
var cl= new Widget(w, Widget.LABEL, 1, 30, 1, 7, "Programs:");
var cb= new Widget(w, Widget.COMBOBOX, 31, 61, 1, 7);
// Add WidgetItems to Combobox
for (i=0; i<items.length; i++)
    var wi = new WidgetItem(cb, items[i]);
// A combobox with many items showing a slider.
var li= new Widget(w, Widget.LABEL, 1, 30, 8, 14, "Long list:");
var ci= new Widget(w, Widget.COMBOBOX, 31, 61, 8, 14);
// Add WidgetItems to Combobox
// As an example we also make some of the WidgetItems unselectable and
// change the background colour
for (i=1; i<=100; i++)
{
    var wi = new WidgetItem(ci, "Item "+i);
    if ( (i % 10) == 5)
    {
        wi.selectable = false;
        wi.background = Widget.WHITE;
    }
}
var e = new Widget(w, Widget.BUTTON, 1, 21, 15, 21, "Exit");
// Assign callbacks
cb.onClick = clicked;
cb.onChange = changed;
ci.onClick = clicked;
ci.onChange = changed;
e.onClick = confirm_exit
// Show the window and start event loop
w.Show();
////////////////////////////////////
function clicked()
{
    // If combobox is clicked then print the current selection
    if (this.selectedItem)
        Message("selection is currently '"+this.selectedItem.text+"'");
}
////////////////////////////////////
function changed()
{
    // If combobox selection is changed then print the new selection
    if (this.selectedItem)
        Message("selection is now '"+this.selectedItem.text+"'");
}
////////////////////////////////////
function confirm_exit()
{
    // Map confirm box
    var ret = Window.Question("Confirm exit", "Are you sure you want to quit?");
    // If the user has answered yes then exit from the script.
    if (ret == Window.YES) Exit();
}
```

See the documentation below and the [Window](#) and [Widget](#) classes for more details.

## Constructor

new WidgetItem(widget[[Widget](#)], text[*string*], selectable (optional)[*boolean*])

## Description

Create a new [WidgetItem](#) object.

## Arguments

Name	Type	Description
widget	<a href="#">Widget</a>	<a href="#">Widget</a> that widget item will be created in. This can be null in which case the <a href="#">WidgetItem</a> will be created but not assigned to a <a href="#">Widget</a> . It can be assigned later by using <a href="#">Widget.AddItem()</a> .
text	string	Text to show on widget item
selectable (optional)	boolean	If the widget item can be selected. If omitted the widget item will be selectable.

## Return type

[WidgetItem](#) object

## Window class

The Window class allows you to create windows for a graphical user interface. [More...](#)

### Class functions

- [BottomBorder\(\)](#)
- [Error](#)(title[*string*], error[*string*], buttons (optional)[*constant*])
- [GetDirectory](#)(initial (optional)[*string*])
- [GetFile](#)(extension (optional)[*string*], allow new (optional)[*boolean*], initial (optional)[*string*])
- [GetFilename](#)(title[*string*], message[*string*], extension (optional)[*string*], initial (optional)[*string*])
- [GetFiles](#)(extension (optional)[*string*])
- [GetInteger](#)(title[*string*], message[*string*], initial (optional)[*integer*])
- [GetNumber](#)(title[*string*], message[*string*], initial (optional)[*float*])
- [GetString](#)(title[*string*], message[*string*], initial (optional)[*string*])
- [Information](#)(title[*string*], info[*string*], buttons (optional)[*constant*])
- [MasterResolution\(\)](#)
- [Message](#)(title[*string*], message[*string*], buttons (optional)[*constant*])
- [Question](#)(title[*string*], question[*string*], buttons (optional)[*constant*])
- [RightBorder\(\)](#)
- [TopBorder\(\)](#)
- [UpdateGUI\(\)](#)
- [Warning](#)(title[*string*], warning[*string*], buttons (optional)[*constant*])

### Member functions

- [Delete\(\)](#)
- [Hide\(\)](#)
- [Recompute\(\)](#)
- [Redraw\(\)](#)
- [Show](#)(modal (optional)[*boolean*])

### Window constants

Name	Description
Window.CANCEL	Show CANCEL button
Window.NO	Show NO button
Window.NONMODAL	Allow <a href="#">Window.Error</a> , <a href="#">Window.Question</a> , <a href="#">Window.Warning</a> etc windows to be non modal
Window.OK	Show OK button
Window.YES	Show YES button

### Constants for Resizing/positioning

Name	Description
Window.BOTTOM	Bottom resizing/positioning of window
Window.CENTRE	Centre (horizontal) positioning of window
Window.LEFT	Left resizing/positioning of window
Window.MIDDLE	Middle (vertical) positioning of window
Window.RIGHT	Right resizing/positioning of window
Window.TOP	Top resizing/positioning of window

### Window properties

Name	Type	Description
------	------	-------------

active	boolean	If true (default) then the window then the window is active and widgets in the window can be used. If false then the window is inactive and the widgets cannot be used.
background	constant	Window background colour. Can be: <a href="#">Widget.BLACK</a> , <a href="#">Widget.WHITE</a> , <a href="#">Widget.RED</a> , <a href="#">Widget.GREEN</a> , <a href="#">Widget.BLUE</a> , <a href="#">Widget.CYAN</a> , <a href="#">Widget.MAGENTA</a> , <a href="#">Widget.YELLOW</a> , <a href="#">Widget.DARKRED</a> , <a href="#">Widget.DARKGREEN</a> , <a href="#">Widget.DARKBLUE</a> , <a href="#">Widget.GREY</a> , <a href="#">Widget.DARKGREY</a> , <a href="#">Widget.LIGHTGREY</a> or <a href="#">Widget.DEFAULT</a>
bottom	float	bottom coordinate of window in range 0.0 (bottom) to 1.0 (top)
height	float	height of window
keepOnTop	boolean	If true then the window will be kept "on top" of other windows. If false (default) then the window stacking order can be changed.
left	float	left coordinate of window in range 0.0 (left) to 1.0 (right)
onAfterShow	function	Function to call <b>after</b> a Window is shown. The Window object is accessible in the function using the 'this' keyword. This may be useful to ensure that certain actions are done after the window is shown. It can also be used to show another window so this enables multiple windows to be shown. To unset the function set the property to null.
onBeforeShow	function	Function to call <b>before</b> a Window is shown. The Window object is accessible in the function using the 'this' keyword. This may be useful to ensure that buttons are shown/hidden etc before the window is shown. Note that it cannot be used to show another window. Use <a href="#">onAfterShow</a> for that. To unset the function set the property to null.
onClose	function	Function to call when a Window is closed by pressing the X on the top right of the window. The Window object is accessible in the function using the 'this' keyword. To unset the function set the property to null.
resize	constant	Window resizing. By default when a Window is shown it is allowed to resize on all sides (left, right, top and bottom) to try to make enough room to show the <a href="#">Widgets</a> . The behaviour can be changed by using this property. It can be any combination (bitwise OR) of <a href="#">Window.LEFT</a> , <a href="#">Window.RIGHT</a> , <a href="#">Window.TOP</a> or <a href="#">Window.BOTTOM</a> or 0. Note that when <a href="#">Window.Show</a> is called this property is set to 0 (i.e. not to resize on any side).
right	float	right coordinate of window in range 0.0 (left) to 1.0 (right)
showClose	boolean	If true (default) then a close (X) button will automatically be added on the top right of the window. If false then no close button will be shown.
shown (read only)	boolean	true if window is currently shown, false if not
title	string	Window title
top	float	top coordinate of window in range 0.0 (bottom) to 1.0 (top)
width	float	width of window

## Detailed Description

The Window class allows you to make windows that you can place [Widgets](#) in to create a graphical user interface. The Widget class also gives a number of static methods for convenience. e.g. [Window.GetInteger\(\)](#). The following very simple example displays some text in a window with a button that unmaps the window when it is pressed and the user confirms that (s)he wants to exit.

```
// Create window with title "Text" from 0.8-1.0 in x and 0.5-0.6 in y
var w = new Window("Text", 0.8, 1.0, 0.5, 0.6);
// Create label widget
var l = new Widget(w, Widget.LABEL, 1, 40, 1, 7, "Press OK to exit");
// Create button widget
var e = new Widget(w, Widget.BUTTON, 11, 30, 8, 14, "OK");
// Assign the onClick callback method to the function confirm_exit'
e.onClick = confirm_exit;
// Show the widget and start event loop
w.Show();
////////////////////////////////////
function confirm_exit()
{
// Map confirm window
var ret = Window.Question("Confirm exit", "Are you sure you want to quit?");
// If the user has answered Yes then exit.
if (ret == Window.YES) w.Exit();
}
```

See the documentation below and the [Widget](#) class for more details.

## Constructor

`new Window(title[string], left[float], right[float], bottom[float], top[float])`

## Description

Create a new [Window](#) object.

## Arguments

Name	Type	Description
title	string	Window title to show in title bar
left	float	left coordinate of window in range 0.0 (left) to 1.0 (right)
right	float	right coordinate of window in range 0.0 (left) to 1.0 (right)
bottom	float	bottom coordinate of window in range 0.0 (bottom) to 1.0 (top)
top	float	top coordinate of window in range 0.0 (bottom) to 1.0 (top)

## Return type

[Window](#) object

## Example

To create a Window 'Example' in the top right half of the screen:

```
var w = new Window('Example', 0.5, 1.0, 0.5, 1.0);
```

## Details of functions

### BottomBorder() [static]

#### Description

Returns the position of the bottom border (in range 0-1). This can be used to help position windows on the screen.

#### Arguments

No arguments

#### Return type

float in range 0-1

#### Example

To obtain the position of the bottom border:

```
var b = Window.BottomBorder();
```

---

### Delete()

#### Description

Deletes the window from T-HIS and returns any memory/resources used for the window. **This function should not normally need to be called.** However, in exceptional circumstances if a script recreates windows many times T-HIS may run out of USER objects on Microsoft Windows because of the way T-HIS creates and shows windows. To avoid this problem this method can be used to force T-HIS to return the resources for a window. **Do not use the Window object after calling this method.**

#### Arguments

No arguments

#### Return type

No return value

#### Example

To delete window w:

```
w.Delete();
```

---

### Error(title[*string*], error[*string*], buttons (optional)[*constant*]) [static]

#### Description

Show an error message in a window.

---



## Arguments

Name	Type	Description
title	string	Title for window.
error	string	Error message to show in window.
buttons (optional)	constant	The buttons to use. Can be bitwise OR of <a href="#">Window.OK</a> , <a href="#">Window.CANCEL</a> , <a href="#">Window.YES</a> or <a href="#">Window.NO</a> . If this is omitted an OK button will be used. By default the window will be modal. If <a href="#">Window.NONMODAL</a> is also given the window will be non-modal instead.

## Return type

Button pressed

## Example

To show error *Critical error!\nAbort?* in window with title *Error* with Yes and No buttons:

```
var answer = Window.Error("Error", "Critical error!\nAbort?", Window.YES |
Window.NO);
if (answer == Window.YES) Exit();
```

---

## GetDirectory(initial (optional)[string]) [static]

## Description

Map the directory selector box native to your machine, allowing you to choose a directory. On Unix this will be a Motif selector. Windows will use the standard windows directory selector.

## Arguments

Name	Type	Description
initial (optional)	string	Initial directory to start from.

## Return type

directory (string), (or null if cancel pressed).

## Example

To select a directory:

```
var dir = Window.GetDirectory();
```

---

## GetFile(extension (optional)[string], allow new (optional)[boolean], initial (optional)[string]) [static]

## Description

Map a file selector box allowing you to choose a file. See also [Window.GetFiles\(\)](#) and [Window.GetFilename\(\)](#).

## Arguments

Name	Type	Description
extension (optional)	string	Extension to filter by.
allow new (optional)	boolean	Allow creation of new file.
initial (optional)	string	Initial directory to start from.

## Return type

filename (string), (or null if cancel pressed).

## Example

To select a file using extension '.key':

```
var file = Window.GetFile(".key");
```

---

**GetFilename**(title[*string*], message[*string*], extension (optional)[*string*], initial (optional)[*string*]) [static]

## Description

Map a window allowing you to input a filename (or select it using a file selector). OK and Cancel buttons are shown. See also [Window.GetFile\(\)](#).

## Arguments

Name	Type	Description
title	string	Title for window.
message	string	Message to show in window.
extension (optional)	string	Extension to filter by.
initial (optional)	string	Initial value.

## Return type

filename (string), (or null if cancel pressed).

## Example

To create an file input window with title *Choose file* and message *Choose the file to open* and return the filename input:

```
var filename = Window.GetFilename("Choose file", "Choose the file to open");
```

---

**GetFiles**(extension (optional)[*string*]) [static]

## Description

Map a file selector box allowing you to choose multiple files. See also [Window.GetFile\(\)](#) and [Window.GetFilename\(\)](#).

## Arguments

Name	Type	Description
extension (optional)	string	Extension to filter by.

## Return type

Array of filenames (strings), or null if cancel pressed.

### Example

To select multiple files using extension '.key':  

```
var files = Window.GetFiles(".key");
```

---

**GetInteger**(title[*string*], message[*string*], initial (optional)[*integer*]) [static]

## Description

Map a window allowing you to input an integer. OK and Cancel buttons are shown.

### Arguments

Name	Type	Description
title	string	Title for window.
message	string	Message to show in window.
initial (optional)	integer	Initial value.

## Return type

value input (integer), or null if cancel pressed.

### Example

To create an input window with title *Input* and message *Input integer* and return the value input:  

```
var value = Window.GetInteger("Input", "Input integer");
```

---

**GetNumber**(title[*string*], message[*string*], initial (optional)[*float*]) [static]

## Description

Map a window allowing you to input a number. OK and Cancel buttons are shown.

### Arguments

Name	Type	Description
title	string	Title for window.
message	string	Message to show in window.
initial (optional)	float	Initial value.

## Return type

value input (float), or null if cancel pressed.

### Example

To create an input window with title *Input* and message *Input number* and return the value input:  

```
var value = Window.GetNumber("Input", "Input number");
```

GetString(title[*string*], message[*string*], initial (optional)[*string*]) [static]

### Description

Map a window allowing you to input a string. OK and Cancel buttons are shown.

### Arguments

Name	Type	Description
title	string	Title for window.
message	string	Message to show in window.
initial (optional)	string	Initial value.

### Return type

value input (string), or null if cancel pressed.

### Example

To create an input window with title *Input* and message *Input string* and return the value input:  

```
var value = Window.GetString("Input", "Input string");
```

---

## Hide()

### Description

Hides (unmaps) the window.

### Arguments

No arguments

### Return type

No return value

### Example

To hide window w:  

```
w.Hide();
```

---

Information(title[*string*], info[*string*], buttons (optional)[*constant*]) [static]

### Description

Show information in a window.

## Arguments

Name	Type	Description
title	string	Title for window.
info	string	Information to show in window.
buttons (optional)	constant	The buttons to use. Can be bitwise OR of <a href="#">Window.OK</a> , <a href="#">Window.CANCEL</a> , <a href="#">Window.YES</a> or <a href="#">Window.NO</a> . If this is omitted an OK button will be used. By default the window will be modal. If <a href="#">Window.NONMODAL</a> is also given the window will be non-modal instead.

## Return type

Button pressed

### Example

To show information *Information* in window with title *Example* with OK and Cancel buttons:

```
var answer = Window.Information("Example", "Information", Window.OK |
Window.CANCEL);
if (answer == Window.CANCEL) Message("You pressed the Cancel button");
```

## MasterResolution() [static]

### Description

Returns the resolution of the master programme window in pixels

### Arguments

No arguments

### Return type

Array containing x and y resolution in pixels

### Example

To get the resolution of the main window:

```
var res = Window.MasterResolution();
```

## Message(title[*string*], message[*string*], buttons (optional)[*constant*]) [static]

### Description

Show a message in a window.

### Arguments

Name	Type	Description
title	string	Title for window.
message	string	Message to show in window.
buttons (optional)	constant	The buttons to use. Can be bitwise OR of <a href="#">Window.OK</a> , <a href="#">Window.CANCEL</a> , <a href="#">Window.YES</a> or <a href="#">Window.NO</a> . If this is omitted an OK button will be used By default the window will be modal. If <a href="#">Window.NONMODAL</a> is also given the window will be non-modal instead.

## Return type

Button pressed

### Example

To show message *Press YES or NO* in window with title *Example* with YES and NO buttons:

```
var answer = Window.Message("Example", "Press YES or NO", Window.YES |
Window.NO);
if (answer == Window.NO) Message("You pressed No");
```

---

## Question(title[*string*], question[*string*], buttons (optional)[*constant*]) [static]

### Description

Show a question in a window.

### Arguments

Name	Type	Description
title	string	Title for window.
question	string	Question to show in window.
buttons (optional)	constant	The buttons to use. Can be bitwise OR of <a href="#">Window.OK</a> , <a href="#">Window.CANCEL</a> , <a href="#">Window.YES</a> or <a href="#">Window.NO</a> . If this is omitted Yes and No button will be used. By default the window will be modal. If <a href="#">Window.NONMODAL</a> is also given the window will be non-modal instead.

### Return type

Button pressed

### Example

To show question *Do you want to continue?* in window with title *Question*:

```
var answer = Window.Question("Question", "Do you want to continue?");
if (answer == Window.NO) Message("You pressed No");
```

---

## Recompute()

### Description

Recomputes the positions of widgets in the window. If you have [static](#) widgets and 'normal' widgets in a window and you show and/or hide widgets the window needs to be recomputed to refresh the graphics, scroll bars etc. Calling this method will recompute and redraw the window.

### Arguments

No arguments

### Return type

No return value

### Example

To recompute window w:

```
w.Recompute();
```

---

## Redraw()

### Description

Redraws the window. Sometimes if you [show](#), [hide](#) or draw graphics on [widgets](#) the window needs to be redrawn to refresh the graphics. Calling this method will redraw the window refreshing the graphics.

### Arguments

No arguments

### Return type

No return value

### Example

To redraw window w:  
`w.Redraw();`

---

## RightBorder() [static]

### Description

Returns the position of the right border (in range 0-1). This can be used to help position windows on the screen.

### Arguments

No arguments

### Return type

float in range 0-1

### Example

To obtain the position of the right border:  
`var b = Window.RightBorder();`

---

## Show(modal (optional))[boolean]

### Description

Shows (maps) the window and waits for user input.

### Arguments

Name	Type	Description
modal (optional)	boolean	If this window is modal (true) then the user is blocked from doing anything else in T-HIS until this window is dismissed). If non-modal (false) then the user can still use other functions in T-HIS. If omitted the window will be modal. Note that making a window modal will stop interaction in all other windows and may prevent operations such as picking from working in any macros that are run from scripts.

## Return type

No return value

### Example

To show window w:

```
w.Show();
```

To show window w allowing the user to use other functions in T-HIS:

```
w.Show(false);
```

---

## TopBorder() [static]

### Description

Returns the position of the top border (in range 0-1). This can be used to help position windows on the screen.

### Arguments

No arguments

### Return type

float in range 0-1

### Example

To obtain the position of the top border:

```
var b = Window.TopBorder();
```

---

## UpdateGUI() [static]

### Description

Force GUI to be updated. This function is not normally needed but if you are doing a computationally expensive operation and want to update the GUI it may be necessary as the GUI update requests are cached until there is spare time to update them. Calling this function forces any outstanding requests to be flushed.

### Arguments

No arguments

### Return type

No return value

### Example

To force update of GUI:

```
Window.UpdateGUI();
```

---

## Warning(title[*string*], warning[*string*], buttons (optional)[*constant*]) [static]

### Description

Show a warning message in a window.

---



---

## Arguments

Name	Type	Description
title	string	Title for window.
warning	string	Warning message to show in window.
buttons (optional)	constant	The buttons to use. Can be bitwise OR of <a href="#">Window.OK</a> , <a href="#">Window.CANCEL</a> , <a href="#">Window.YES</a> or <a href="#">Window.NO</a> . If this is omitted an OK button will be used. By default the window will be modal. If <a href="#">Window.NONMODAL</a> is also given the window will be non-modal instead.

## Return type

Button pressed

## Example

To show warning *Title is blank\nSet to ID?* in window with title *Warning* with Yes and No buttons:

```
var answer = Window.Warning("Warning", "Title is blank\nSet to ID?", Window.YES  
| Window.NO);  
if (answer == Window.NO) Message("You pressed No");
```

---



## APPENDIX K - Typed Commands

### K.1 Global Menu

<b>PL</b> - Plot	<b>CL</b> - Clear Screen
<b>ZM</b> - Zoom	<b>AU</b> - Auto Scale Plot
<b>CE</b> - Centre	<b>PT</b> - Point on Screen
<b>PF</b> - Write Postscript file (use default)	
<b>PC</b> - Write Postscript file (Colour)	
<b>PB</b> - Write Postscript file (Blank/White)	
<b>BL</b> - Blank Curve	<b>UB</b> - Unblank Curve
<b>RM</b> - Remove a Curve	<b>ER</b> - Erase all curves
<b>GS</b> - Global Status	<b>CO</b> - Condense Curves
<b>Y1</b> - 1st Y axis	<b>Y2</b> - Second Y axis
<b>DOU</b> - Double Y axis (ON/ OFF)	
<b>CF</b> - Command file (read)	<b>SF</b> - Session file (write)
<b>CS</b> - Close session file	
<b>EX</b> - Exit	
<b>!</b> - Backspace	<b>/</b> - Top level menu
<b>Q</b> - Abort operation	
<b>;</b> - End of command string	

### K.2 List Commands

<b>LS</b> - List all files in current directory	<b>LC</b> - List all files "*.cur" in current directory
<b>LB</b> - List all files "*.bdf" in current directory	<b>LK</b> - List all files "*.key" in current directory
<b>LI</b> - List all files ASCII files in current directory	

<a href="#"><u>GM - Global Menu</u></a>	
---	--

<b>MO</b> - Model options	<b>RE</b> <file> - Read Model Files
	<b>DA</b> - Read Data from model
	<b>GL</b> <component> - Global data
	<b>PA</b> <id> <component> - Part data
	<b>NO</b> <id> <component> - Node data
	<b>SO</b> <id> <component> - Solid data
	<b>BE</b> <id> <component> - Beam data
	<b>SH</b> <id> <component> - Shell data
	<b>TS</b> <id> <component> - Thick Shell data
	<b>WA</b> <id> <component> - Part data
	<b>SPR</b> <id> <component> - Spring data
	<b>SEA</b> <id> <component> - Seatbelt data
	<b>RET</b> <id> <component> - Retractor data
	<b>SL</b> <id> <component> - Slipping data
	<b>CO</b> <id> <component> - Contact data
	<b>REA</b> <id> <component> - Reaction data
	<b>AI</b> <id> <component> - Airbag data
	<b>JO</b> <id> <component> - Joint data
	<b>SEC</b> <id> <component> - Section data
	<b>SU</b> <id> <component> - Subsystem data
	<b>P_G</b> <id> <component> - Part Group data
	<b>G_C</b> <id> <component> - Geometrical Contact data
	<b>RI</b> <id> <component> - Rigid Body data
	<b>SPO</b> <id> <component> - Spotweld data
	<b>SPC</b> <id> <component> - SPC data
	<b>FS</b> <id> <component> - Fluid structural interaction data
	<b>BO</b> <id> <component> - Boundary condition data
	<b>SPH</b> <id> <component> - SPH data
	<b>SE</b> - Select Models
	<b>DE</b> - Delete Models
	<b>LI</b> - List Models
	<b>SU</b> - Set Surface
<b>RE</b> - Read data	<b>CU</b> - Read T/HIS curve file
	<b>CU NO</b> - Read T/HIS curve file (ignore any style definitions)
	<b>BD</b> - Read Bulk data file
	<b>KW</b> - Read from LS-DYNA KEYWORD input file
	<b>KY</b> - Input curve from keyboard
	<b>CSV</b> - Read a CSV file (X,Y,X,Y,X,Y)
	<b>CSV2</b> - Read a CSV file (X,Y, Y,Y,Y,Y)
	<b>ISO</b> - Read ISO curve data (multiple channels)
	<b>ISO2</b> - Read ISO curve data (single channel)
<b>WR</b> - Write options	<b>WR</b> - Write curve file
	<b>WA</b> - Write all curves to a T/HIS curve file
	<b>KEY</b> - Write curves to a LS-DYNA Keyword file
	<b>CSV</b> - Write curves to a CSV file (X,Y,X,Y,X,Y)
	<b>CSV2</b> - Write curves to a CSV file (X,Y, Y,Y,Y,Y)
	<b>LI</b> - List curve data on screen
	<b>RE</b> - Report curve data to file
	<b>SU</b> - Summary of curve
	<b>ST</b> - Status

<b>DE</b> - Defaults	<b>AU</b> - Auto Scaling	<b>ON</b> - Autoscaling on
		<b>OFF</b> - Autoscaling off
		<b>DX</b> - Define new x limits (minimum,maximum)
		<b>XMN</b> - Define new minimum x limit
		<b>XMN</b> - Define new maximum x limit
		<b>DY</b> - Define new y limit (min,max)
		<b>YMN</b> - Define new minimum y limit
		<b>YMX</b> - Define new maximum y limit
		<b>2DY</b> - Define new second y axis limits (min,max)
		<b>YMN2</b> - Define new minimum second y limit
		<b>YMX2</b> - Define new maximum second y limit
		<b>ST</b> - Status
	<b>TI</b> - Title	
	<b>LA</b> - Axes labels (user defined)	<b>AU</b> - Use automatic axes labels (both)
		<b>AX</b> - Use automatic x axis labels
		<b>AY</b> - Use automatic y axis labels
		<b>2AY</b> - Use automatic 2nd y axis labels
		<b>DX</b> - Define new x axis plot label
		<b>DY</b> - Define new y axis plot label
		<b>2DY</b> - Define new 2nd y axis plot label
		<b>ST</b> - Status
	<b>AW</b> - Axis line width	
	<b>AX</b> - Axis types	
	<b>AC</b> - Axis Colour	
	<b>GR</b> - Grid lines	<b>ON</b> - Turn grid on
		<b>OFF</b> - Turn grid off
		<b>AX</b> - Automatic x-axis grid intervals
		<b>AY</b> - Automatic y-axis grid intervals
		<b>MX</b> - Manual x-axis grid intervals
		<b>MY</b> - Manual y-axis grid intervals
		<b>IX</b> - Define x-axis grid intervals
		<b>IY</b> - Define y-axis grid intervals
		<b>OX</b> - Define x-axis grid offset
		<b>OY</b> - Define y-axis grid offset
		<b>TH</b> - Define grid line thickness
	<b>GW</b> - Grid width	
	<b>UL</b> - User Line	
	<b>LL</b> - Line labels	
	<b>MP</b> - Model Prefix	<b>ON</b> - Turn model prefix on
		<b>OFF</b> - Turn model prefix off
		<b>AUTO</b> - Add prefix if more than one model
	<b>PR</b> - Prefix Format	<b>ID</b> - Model ID
		<b>DIR</b> - Model directory
		<b>THE</b> - Root of THE filename
		<b>USER</b> - User defined
	<b>PF</b> - Plot format	
	<b>WX</b> - Window size (x) "pixels"	
	<b>WY</b> - Window size (y) "pixels"	
	<b>RV</b> - Reverse Foregorund / Background	
	<b>FO</b> - Foreground Colour	
	<b>BA</b> - Background Colour	
	<b>CU</b> - Curve through points ON/OFF	
	<b>SY</b> - Symbols ON/OFF	
	<b>BD</b> - Border ON/OFF	
	<b>BW</b> - Border width	
	<b>BC</b> - Border Colour	
	<b>LW</b> - Default line width	
	<b>SMN</b> - Show minimum value	
	<b>SMX</b> - Show maximum value	
	<b>LXMN</b> - Label x value at minimum	
	<b>LYMN</b> - Label y value at minimum	
	<b>LXMX</b> - Label x value at maximum	

	<b>LYMX</b> - Label y value at maximum
	<b>RE</b> - Reset to defaults
	<b>ST</b> - Status
<b>FO</b> - Font	<b>TI</b> <font> <size> <colour> - Title
	<b>XL</b> <font> <size> <colour> - X Axis Label
	<b>XU</b> <font> <size> <colour> - X Axis Units
	<b>YL</b> <font> <size> <colour> - Y Axis Label
	<b>YU</b> <font> <size> <colour> - Y Axis Units
	<b>Y2L</b> <font> <size> <colour> - 2nd Y Axis Label
	<b>Y2U</b> <font> <size> <colour> - 2nd Y Axis Units
	<b>LE</b> <font> <size> <colour> - Curve Legend
	<b>ALL</b> <font> <size> <colour> - All labels
<b>ED</b> <curve ID> - Edit option	<b>F</b> - move Forward next 16 lines
	<b>B</b> - move Back 16 lines
	<b>T</b> - move to Top of curve
	<b>E</b> - move to End of curve
	<b>n(umber)</b> - move to line n
	<b>C n</b> - Change line n
	<b>I n</b> - Insert before line n
	<b>A n</b> - Append after line n
	<b>D n1 n2</b> - Delete from line n1 to n2
	<b>L</b> - change Line label
	<b>R</b> - Reset edited curve back to original
	<b>W or S</b> - write curve
	<b>PE</b> - Plot Edited curve
	<b>PA</b> - Plot Edited And original curve
	<b>PL</b> - PLot stored T/HIS curves
	<b>Q</b> - Quit the editor
<b>OP</b> - Operate	<b>ADX/Y</b> - Add
	<b>MUX/Y</b> - Multiply
	<b>SUX/Y</b> - Subtract
	<b>DIX/Y</b> - Divide
	<b>CAT</b> - Concatenate 2 curves
	<b>MAP</b> - Map one curve onto another
	<b>COM</b> - Combine curves
	<b>ERR</b> - Error functions
	<b>INT</b> - Integrate
	<b>DIF</b> - Differentiate
	<b>SMO</b> - Smooth
	<b>LSQ</b> - Least squares fit
	<b>SQR</b> - Square root
	<b>NOR</b> - Normalise
	<b>REC</b> - Reciprocal
	<b>ABS</b> - Absolute values
	<b>TRA</b> - Translate
	<b>REV</b> - Reverse
	<b>CLP</b> - Clip
	<b>ZERO</b> - Translate the curve to (0,0)
	<b>ORDER</b> - Reverse the order of the curve points
	<b>VEC</b> - Vector magnitude
	<b>VEC2</b> - Vector Magnitude (2D)
	<b>SUM</b> - Sum of 'n' curves
	<b>ENV</b> - Envelope of 'n' curves
	<b>MIN</b> - Minimum of 'n' curves
	<b>MAX</b> - Maximum of 'n' curves
	<b>AVE</b> - Average of 'n' curves
	<b>R-AV</b> - Rolling Average of 'n' curves
	<b>STR</b> - Convert stress/strain curve

<b>AM</b> - Automotive options	<b>C60</b> - Class 60 filter
	<b>C180</b> - Class 180 filter
	<b>C600</b> - Class 600 filter
	<b>C1000</b> - Class 100 filter
	<b>BUT</b> - Butterworth filter
	<b>FIR</b> - FIR filter
	<b>HIC</b> - HIC value
	<b>HICD</b> - HIC(d) value
	<b>CLI</b> - 3ms Clip value
	<b>EXC</b> - Exceedence Plot
	<b>VC</b> - Viscous Criteria (ECER95)
	<b>VC2</b> - Viscous Criteria (IIHS)
	<b>ASI</b> - Acceleration Severity Index (BS EN 1317-1:1998)
	<b>ASI2</b> - Acceleration Severity Index (BS EN 1317-1:2010)
	<b>THIV</b> - Theoretical Head Impact Velocity
	<b>NIJ</b> - Neck Injury
	<b>TTI</b> - Thoracic Trauma Index
	<b>NOR</b> - Normalise
	<b>REG</b> - Regularise
	<b>VEC</b> - Vector Magnitude
	<b>VEC2</b> - Vector Magnitude (2D)
	<b>ACU</b> - Airbag Control Unit
<b>MA</b> - Maths operations	<b>SQRT</b> - Square Root
	<b>LOG</b> - Natural Log
	<b>EXP</b> - e to power of
	<b>LOG10</b> - Log to base 10
	<b>**</b> - To raise to power
	<b>SIN</b> - Sine
	<b>COS</b> - Cosine
	<b>TAN</b> - Tangent
	<b>ASIN</b> - Arc sine
	<b>ACOS</b> - Arc cosine
<b>SE</b> - Seismic options	<b>ATAN</b> - Arc tangent
	<b>DV</b> - Displacement to velocity spectra
	<b>DA</b> - Displacement to acceleration spectra
	<b>VD</b> - Velocity to displacement spectra
	<b>VA</b> - Velocity to acceleration spectra
	<b>AD</b> - Acceleration to displacement spectra
	<b>AV</b> - Acceleration to velocity spectra
	<b>DS</b> - Produce a design spectrum from a response spectrum
	<b>RS</b> - Produce response spectra from input accelerations
<b>UT</b> - Utility functions	<b>EFT</b> - Fast fourier transformation
	<b>CL</b> - Colour laser output
	<b>GL</b> - Greyscale laser output
	<b>LW</b> - Line width
	<b>SA</b> - Solid axes (x=0 & y=0 axes solid)
<b>ST</b> - Line styles	<b>RE</b> - Read in style file
	<b>WR</b> - Write out style file
	<b>DE</b> - Reset styles to default settings
	<b>SET</b> - Set a T/HIS line style
	<b>FIX</b> - Turn fix line styles on/off
<b>HE</b> - Help	
<b>CU</b> - Curve editing options	<b>LA</b> - Set a new curve label
	<b>TI</b> - Set a new curve title
	<b>XL</b> - Set a new curve x-axis label
	<b>YL</b> - Set a new curve y-axis label
	<b>TA</b> - Set a new curve tag
<b>GRO</b> - Group options	<b>READ</b> - Read a T/HIS group file
	<b>LIST</b> - List all T/HIS groups
	<b>DELETE</b> - Delete all T/HIS groups
	<b>CREATE</b> - Create a new T/HIS group

<b>IM</b> - Image output options	<b>JPEG &lt;file&gt;</b> - Capture a JPEG image
	<b>BMP_U &lt;file&gt;</b> - Capture an uncompressed Bitmap image
	<b>BMP_C &lt;file&gt;</b> - Capture a compressed Bitmap image
	<b>PPM &lt;file&gt;</b> Capture a portable pixmap file
<b>PREF</b> - Define T/HIS user preferences	<b>REG</b> - Set time interval for automatic curve resampling
	<b>CONV</b> - Set/unset automatic conversion from ms to s when filtering
	<b>FILE</b> - Turn on/off output of injury criteria values and error calculations to ASCII files
	<b>SHOW</b> - Turn on/off display of HIC/ 3ms clip values
	<b>ZERO</b> - Turn on/off automatic creation of (0,0) point when reading data from ASCII files



# Installation organisation

The version 12 installation can be customised to try and avoid a number of issues that often occur in large organisations with many users.

- Large organisations generally imply large networks, and it is often the case that the performance of these networks can be intermittent or poor, therefore it is common practice to perform an installation of the software on the local disk of each machine, rather than having a single installation on a remote disk.

This avoids the pauses and glitches that can occur when running executable files over a network, but it also means that all the configuration files in, or depending upon, the top level "Admin" directory have to be copied to all machines and, more to the point, any changes or additions to such files also have to be copied to all machines.

- In larger organisations the "one person per computer" philosophy may not apply, with the consequence that users will tend to have a floating home area on a network drive and may not use the same machine every day.

This is not usually a problem on Linux where the "home" directory is tied to the login name not the machine. However on Windows platforms it means that %USERPROFILE%, which is typically on the local C drive of a machine, is not a good place to consider as "home" since it will be tied to a given computer, therefore a user who saves a file in his home directory on machine A may not be able to access it from machine B.

- In a similar vein placing large temporary files on the /tmp partition (Linux) or the C: drive (Windows) may result in local disks becoming too full, or quotas exceeded.

This section gives only a brief summary of the installation organisation, and you should refer to the separate Installation Guide if you want to find out more about the details of installation, licensing, and other related issues.

## Version 12.0 Installation structure

In version 12.0 the option is provided to separate a top-level 'administration' directory from the 'installation' one where the executables are located.

For large installations on many machines this allows central configuration and administration files to exist in one place only, but executables to be installed locally on users' machines to give better performance. Version 12.0 also allows the following items to be configured

- The location for user manuals and other documentation.
- The definition of a user's home directory.
- The definition of the temporary directory for scratch files.

In addition parsing of the 'oa\_pref' (preferences) file will now handle environment variables, so that a generic preference can be configured to give a user-specific result, and preferences may be 'locked' so that those set at the administration level cannot be changed by users.

These changes are entirely optional, and users performing a simple installation on a single machine do not need to make any changes to their existing installation practice.

Directory	Status	Directory Content and purpose	oa_pref file option
OA_ADMIN_xx	<i>Optional</i>	Top level configuration files. (xx =12 for release 12.0, thus OA_ADMIN_12)  Admin level oa_pref file Other configuration files Timeout configuration file	

<b>OA_ADMIN</b>	<i>Optional</i>	Same as <b>OA_ADMIN_12</b> , provided for backwards compatibility with earlier releases.  It is recommended that plain <b>OA_ADMIN</b> , without the <b>_xx</b> version suffix, is not used since otherwise there is no easy way of distinguishing between parallel installations of different releases of the Oasys Ltd software in an installation.  <i>If <b>OA_ADMIN_12</b> is not defined then this non-release specific version is checked.</i>	
<b>OA_INSTALL_xx</b>	<i>Optional</i>	( <b>xx</b> =12 for release 12.0, thus <b>OA_ADMIN_12</b>  All executables Installation level oa_pref file	<b>oasys*install_dir:</b> <b>&lt;pathname&gt;</b>
<b>OA_INSTALL</b>	<i>Optional</i>	Same as <b>OA_INSTALL_12</b> .  If no " <b>OA_ADMIN_xx</b> " directory is used and all software is simply placed in this "install" directory, which would be typical of a single-user installation, then it is recommended that the <b>_xx</b> version suffix is used in order to keep parallel installations of different releases of the Oasys Ltd software separate on the machine.  <i>If <b>OA_INSTALL_12</b> is not defined then this non-release specific version is checked</i>	<b>oasys*install_dir:</b> <b>&lt;pathname&gt;</b>
<b>OA_MANUALS</b>	<i>Optional</i>	Specific directory for user manuals. If not defined then will search in: <b>OA_ADMIN_xx/manuals</b> (xx = major version number) <b>OA_INSTALL/manuals</b>	<b>oasys*manuals_dir:</b> <b>&lt;pathname&gt;</b>
<b>OA_HOME</b>	<i>Optional</i>	Specific "home" directory for user when using Oasys Ltd software. If not defined will use: <b>\$HOME</b> (Linux) <b>%USERPROFILE%</b> (Windows)	<b>oasys*home_dir:</b> <b>&lt;pathname&gt;</b>
<b>OA_TEMP</b>	<i>Optional</i>	Specific "temporary" directory for user when using Oasys Ltd software. If not defined will use: <b>P_tmpdir</b> (Linux, typically /tmp) <b>%TEMP%</b> (Windows, typically C:\temp)	<b>oasys*temp_dir:</b> <b>&lt;pathname&gt;</b>

It will be clear from the table above that no Environment variables have to be set, and that all defaults will revert to pre-9.4 behaviour. In other words users wishing to keep the status quo will find behaviour and layout unchanged if they do nothing.

#### **OA\_INSTALL\_xx**

Previously the software used the **OA\_INSTALL** (renamed from **OASYS**) environment variable to locate the directory the software was installed in.

- On Windows this is no longer required as the software can work out its own installation directory. As this environment variable is no longer required it is recommended that it is removed from machines it is currently set on as in some cases where more than one version has been installed in different directories it can cause problems.
- On LINUX systems the "oasys\_12" script that starts the SHELL automatically sets this Environment Variable and passes it to any application started from the SHELL. If you run applications directly from the command line and bypass the SHELL then you should set **OA\_INSTALL\_xx** so that the software can locate manuals and other required files.

#### **OA\_ADMIN\_xx**

Users wishing to separate configuration and installation directories will be able to do so by making use of the new top level **OA\_ADMIN\_xx** directory.

## Installation Examples

The following diagrams illustrate how the installation might be organised in various different scenarios..

### a) Single user installation on one machine

There is no need to worry about separating administration and installation directories, and the default installation of all files in and below the single installation directory will suffice.

It is suggested that the **xx** version suffix of **OA\_INSTALL\_xx** is used in order to keep parallel installations of different releases of the Oassys Ltd software separate on the machine.

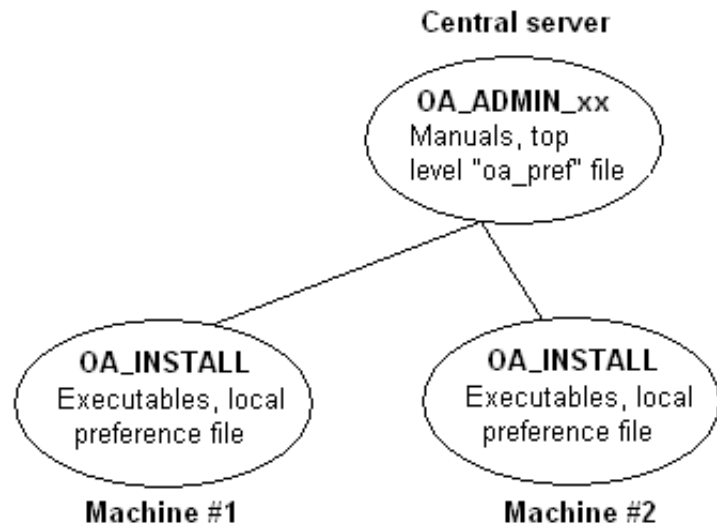


### b) A few machines on a small network, each user has his own machine

The top level administration directory can be installed on a network server, possibly also locating the manuals centrally.

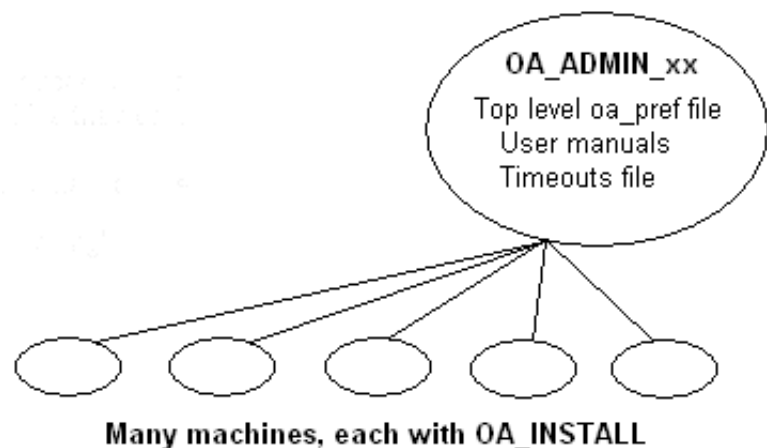
Each user's machine has its own 'installation' directory to give good performance, but there is no need to manage home or temporary directories centrally since each user 'owns' his machine.

If network performance is good an alternative would be to install executables on the central server, meaning that local OA\_INSTALL directories are not required.



### c) Large corporate network

There is no need to worry about separating administration and installation directories, and the default installation of all files in and below the single installation directory will suffice.



## Dynamic configuration using the top level oa\_pref file.

A further improvement is that all environment variables below **OA\_ADMIN\_xx** may either be set explicitly, or dynamically using the options in the oa\_pref file at the top **OA\_ADMIN\_xx** level. This permits parallel installations of different versions of the software to co-exist, with only the top level administration directory names being distinct. For example:

Release 12.0	Release 12.1
Top level directory <b>OA_ADMIN_12</b>	Top level directory <b>OA_ADMIN_121</b>
oa_pref file in <b>OA_ADMIN_12</b> contains:  <b>oasys*install_dir:</b> <pathname for 12.0 installation> <b>oasys*manuals_dir:</b> <pathname for 12.0 manuals>  <b>oasys*home_dir:</b> <pathname for home directory> <b>oasys*temp_dir:</b> <pathname for temporary files>	oa_pref file in <b>OA_ADMIN_121</b> contains:  <b>oasys*install_dir:</b> <pathname for 12.1 installation> <b>oasys*manuals_dir:</b> <pathname for 12.1 manuals>  } would almost certainly be unchanged between major } versions, although they could be different if desired
Pathnames in the oa_pref file may contain environment variables which will be resolved before being applied.	

## The hierarchy of oa\_pref file reading

It will be clear from the above that in a large installation the "oa\_pref" files have a significant role. Each piece of software reads them in the following order:

<b>OA_ADMIN_xx</b>	Top level configuration
<b>OA_INSTALL_xx</b>	Installation level
<b>OA_HOME</b>	User's personal "home" file
Current working directory	File specific to the current directory (rarely used)

The rules for reading these files are:

- If a given directory does not exist, or no file is found in that directory, then no action is taken. This is not an error.
- A more recently read definition supersedes one read earlier, therefore "local" definitions can supersede "global" ones (unless it was locked).
- If two of more of the directories in the table above are the same then that file is only read once from the first instance.

## Locking Preference Options

From version 9.4 onwards preference options can be locked. If a preference option is locked in a file then that preference option will be ignored in any of the subsequent preference files that are read.

Therefore by locking a preference in a top-level file in the hierarchy above, eg in **OA\_ADMIN\_xx**, and then protecting that file to be read-only, an administrator can set preferences that cannot be altered by users since any definitions of that preference in their private oa\_pref files will be ignored.

Preferences are locked by using a hash (#) rather than an asterisk (\*) between the code name and the preference string. For example:

<b>primer*maximise: true</b>	Normal case using "*", means an unlocked preference
------------------------------	---

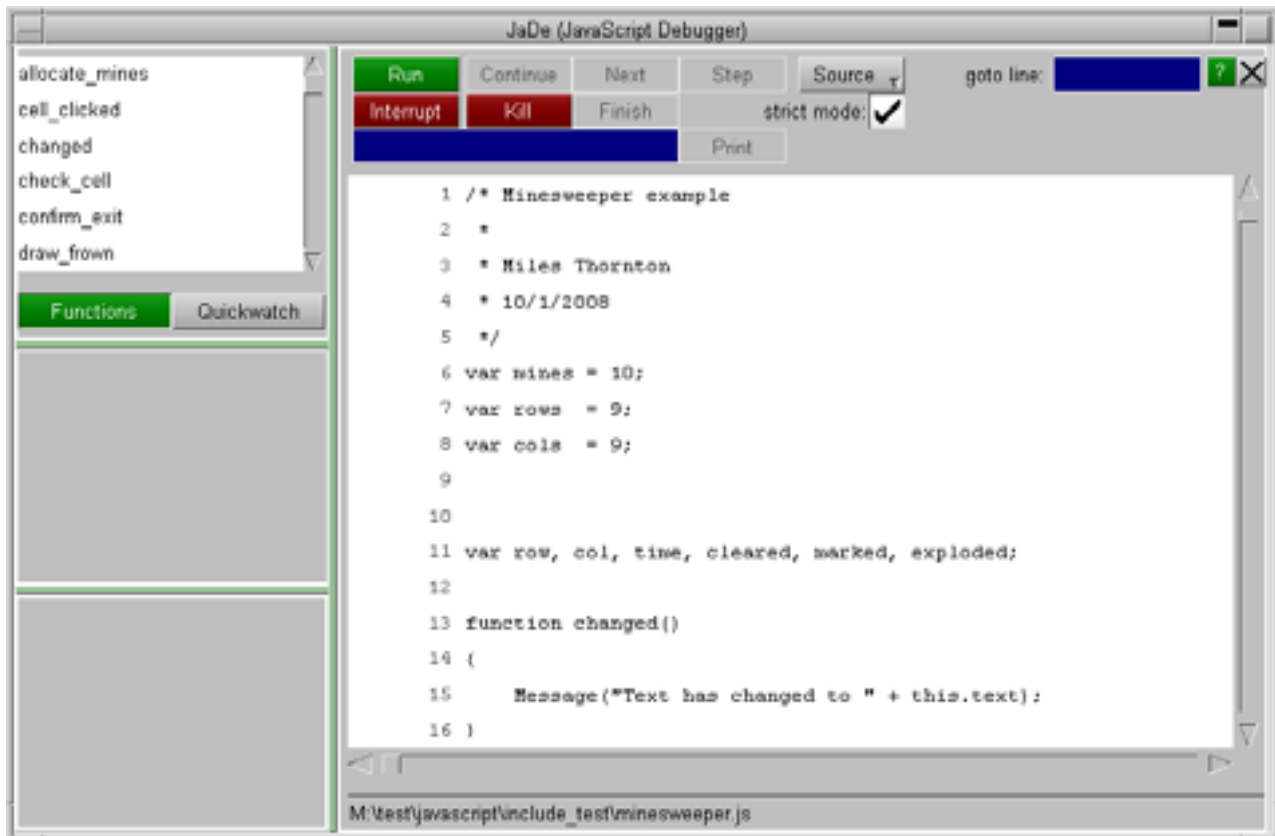
<code>primer#maximise: true</code>	Locked case using "#"
------------------------------------	-----------------------

These changes may be made either by editing the file manually, or by using the preferences editor.



# JaDe: The JavaScript debugger

JaDe is included in D3PLOT, PRIMER and T/HIS to help debug and develop JavaScripts. It is started by selecting a script and pressing the **Debug** button in the JavaScript menu in any of the programs. The initial screen is shown below.



It is fairly basic but hopefully has enough functionality for people to be able to find and fix problems in scripts.

## Viewing the script files and functions

The main part of the window shows the script file. If your script is broken up into separate file (by using Use) then you can get a list of the different files and view them by using the **Source** popup. To go to a particular line in the file use the **goto line** textbox.

A list of the functions in the script is shown in the **Functions** menu on the top left. If you want to look at a particular function then click on the function name and the main text window will jump to the correct file and line.

## Adding/removing breakpoints

A breakpoint is a line in the script where execution will pause in JaDe. To add a breakpoint either left click on the line you want the breakpoint on or right click on the line and select **Create breakpoint** from the popup. A red circle is then drawn on the line to show that there is an active breakpoint.

```
112 function allocate_mines()
113 {
114     var n = mines;
115
116     while (n)
117     {
```

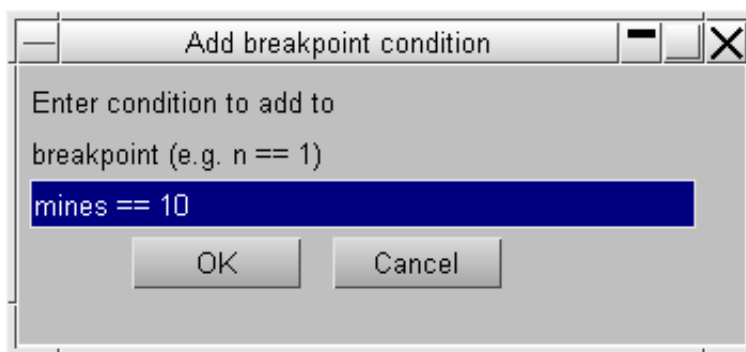
Additionally the breakpoint will also be added to the list in the breakpoint window (bottom left of JaDe). You can click on this at any time and the main text window will jump to the correct file and line.

Active breakpoints are shown with a red circle. Breakpoints can be activated/deactivated by clicking on the line again. Unactive breakpoints are shown as a grey circle instead of a red one. They are also shown in grey text in the breakpoint window.

To delete a breakpoint right click on the line and select **Delete breakpoint**. The breakpoint will be deleted.

## Conditional breakpoints

Sometimes it is useful to only stop at a breakpoint if a certain condition is met. For example in the above example we may only want to stop at line 114 if `mines` is 10. You can do this by right clicking on the the breakpoint and selecting **Add condition**.



A window is mapped allowing you type in the condition you want to try to meet. The condition should be a JavaScript expression which evaluates to true if you want the breakpoint to stop execution, or false if you want the breakpoint to be skipped. In this example the condition is `n == 10`.

If a breakpoint has a condition associated with it a C is drawn on the circle and in the breakpoint window. The condition can be edited again or removed by right clicking on the breakpoint and selecting either **Edit condition** or **Remove condition** from the popup.

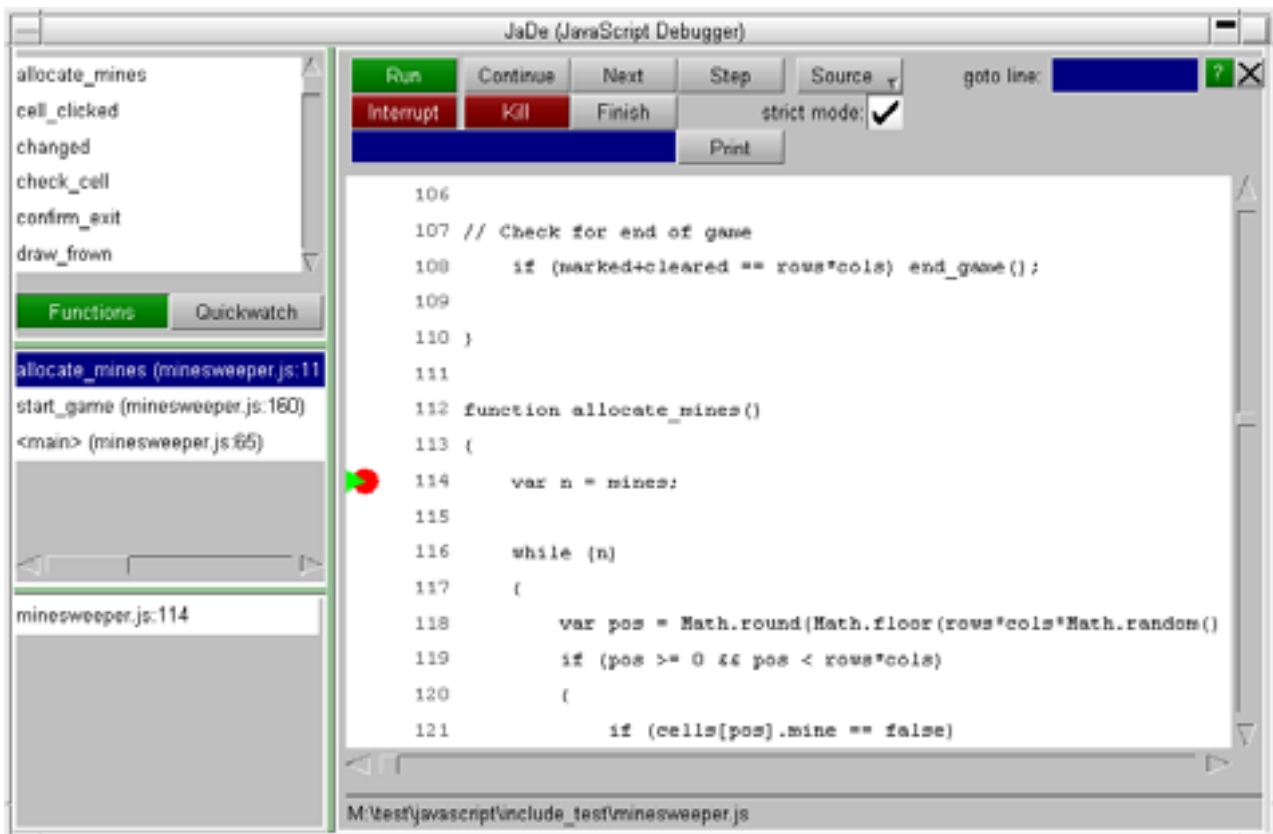
## Running the script

Running the script is controlled by the buttons at the top of the debugger window. By default the script will be run in the debugger in 'strict mode'. This tries to pick up things which you might not have intended by running the script in a stricter environment doing more checking. You can toggle this on/off by using the **strict mode** checkbox.

## Starting and stopping

To start the script press the **Run** button. Execution of the script will start. If you have not defined any breakpoints then the script will run until it finishes (unless there are some script errors or exceptions). If there is a breakpoint then the debugger will stop execution of the script when it reaches it. If the script is running and you want to pause execution of the script at any time you can press **Interrupt**.





The line that the debugger has paused the script on is shown by a green triangle. In the above example it is paused at line 114. The middle panel on the left shows the [call stack](#). See the [call stack section below](#) for more details.

## Stepping and continuing

Once the script is paused in the debugger you can step through the source code by using the **Continue**, **Next**, **Step** and **Finish** buttons.

**Continue** will resume execution of the script again.

**Next** continues to the next line in the current function. i.e. it will step *over* a function call.

**Step** continues execution to the next source line (which may be in a different function. i.e. it will step *into* a function call).

**Finish** will finish executing the current function and stop at the next line in the calling function (the function above this in the [call stack](#)).

Alternatively, if you want to continue until a particular line you can right click on the line you want to continue until and select **Continue to here** from the popup.

## Printing the value of a variable

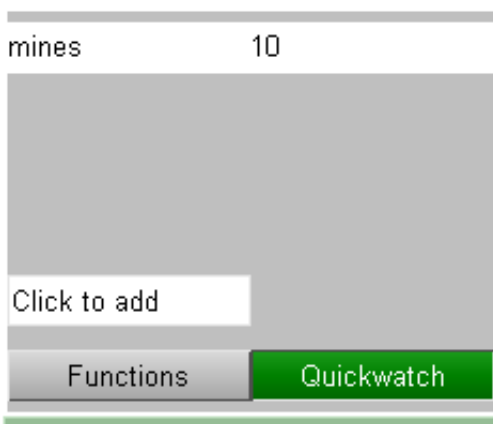
If you want to see the value of a variable you can type the name of the variable you want to see in the textbox at the top of the debugger and press **Print**. JaDe will evaluate the variable and output the result in the statusbar at the bottom of the debugger.

## Using Quickwatch

If you want to look at the values for lots of variables it is annoying to have to type the variable name in and press **Print** for each one. A better way is to use **Quickwatch** at the top left of JaDe



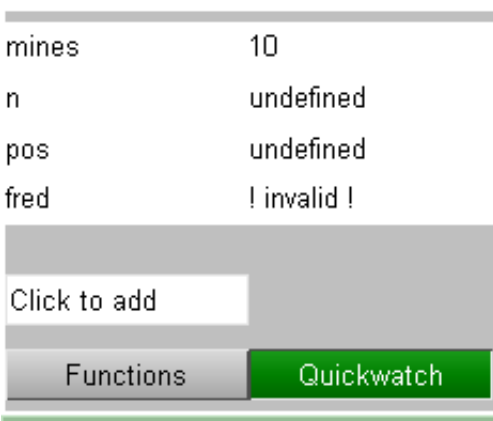
Type the name of the variable that you want to watch in the **Click to add** textbox. A line will be added for the variable showing its name and value. e.g. in the following image the variable `mines` is being displayed and its current value is 10. If the value is very long hover over the value to get the whole string.



You can add any number of variables to watch. To remove one right click on the variable and select **Remove quickwatch** from the popup.

If a variable exists and has been assigned to then the value is displayed. e.g. `mines` in the following example. If the variable exists but it has not yet had a value assigned its value is the `undefined` value. e.g. `pos` in the following example.

If the variable does not exist the value is shown as `! invalid !`. e.g. `fred` in the following example.



## The call stack

The call stack shows which functions have been called in the script to get to the current point. It is the middle left window in JaDe.



The top line shows the function that the script is currently paused at. The other lines show the calling functions in order. The above example can be read as:

1. The script starts
2. On line 65 in script file minesweeper.js in the 'main' program the function `start_game` is called.
3. On line 160 in script file minesweeper.js in function `start_game` the function `allocate_mines` is called
4. On line 114 in script file minesweeper.js in function `allocate_mines` the script is paused.

This information is sometimes very useful in more complicated scripts to find out the order things are done in.

The function that the user is currently looking at is highlighted in blue. You can move up or down the call stack by clicking on a line. The main text window will jump to the correct file and line. The line will be shown with a blue triangle instead of a green triangle.

## Exceptions

Sometimes when developing a script you get errors that you need to try to investigate and fix. e.g. an object is null when it should be defined or you try to call a method that does not exist for an object. In these cases an exception is thrown by JavaScript and the script would terminate if run normally. JaDe will trap the exception and stop at the line where the exception occurred. e.g. If for example you have the following code:

```
var w = new Window('Example', 0.5, 1.0, 0.5, 1.0);  
w.BadMethod();  
w.Show();
```

There is no method called `BadMethod` for a `Window`. JaDe will stop at this point and allow you to look at the script.